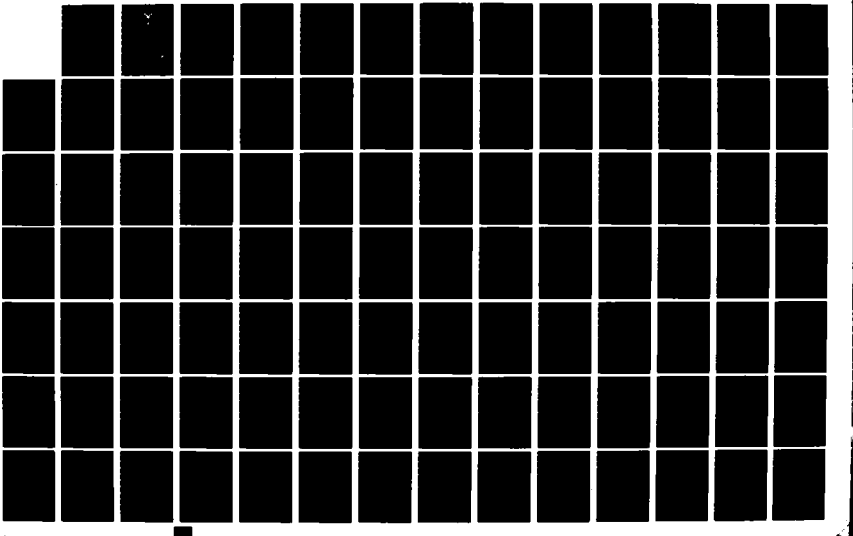
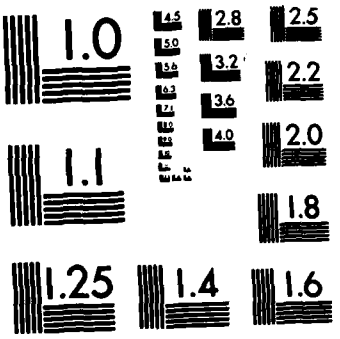
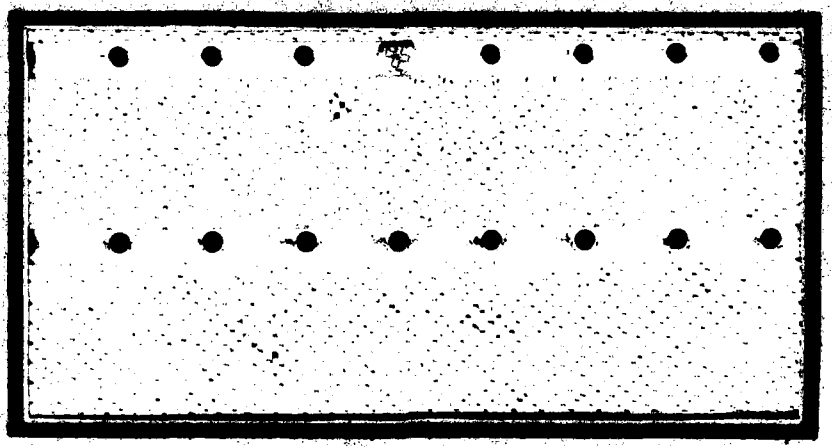
AD-A124 901 EZDRAW AN INTERACTIVE COMPUTER GRAPHICS PROGRAM TO DESIGN BAR LINE OR PIE GRAPHS(U) AIR FORCE INST OF TECH WRIGHT-PATTERSON AFB OH SCHOOL OF ENGI.. R VEA DEC 82 1/3

UNCLASSIFIED AFIT/GE/MA/82D-1 F/G 9/2 NL







DEPARTMENT OF THE AIR FORCE

AIR UNIVERSITY (ATC)

AIR FORCE INSTITUTE OF TECHNOLOGY

Wright-Patterson Air Force Base, Ohio

AFIT/GE/MA/82D-1

EZDRAW
AN INTERACTIVE COMPUTER GRAPHICS PROGRAM TO DESIGN BAR, LINE, OR PIE GRAPHS

THESIS

AFIT/GE/MA/82D-1

ROLDAN VEA
CPT USA

Approved for public release; distribution unlimited.

AFIT/GE/MA/82D-1

EZDRAW-AN INTERACTIVE COMPUTER GRAPHICS PROGRAM TO DESIGN BAR, LINE, OR PIE GRAPHS

THESIS

Presented to the Faculty of the School of Engineering

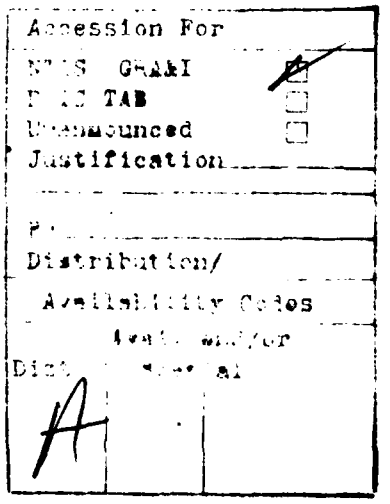
of the Air Force Institute of Technology

Air University

In Partial Fulfillment of the

Requirements for the Degree of

Master of Science



by

Roldan Vea, B.S.

CPT USA

Graduate Electrical Engineering

December 1982

Approved for public release; distribution unlimited.

Preface

Computer graphics is an area of study that interests me. Therefore, when the occasion arose to conduct a thesis research in this area, I readily accepted the challenge and commitment.

The EZDRAW graphics system was initially designed for implementation using TEKLIB, a library of FORTRAN callable subroutines developed by the thesis sponsor. However, during the course of system development, AFWAL purchased the DI-3000 and GRAFMAKER software graphics packages (from Precision Visuals). EZDRAW was therefore redesigned to take advantage of these graphics tools.

At this time, I would like to thank the members of the faculty who served on the thesis evaluation committee. To Professor Charles Richard, my advisor, I express my gratitude for the welcomed guidance, stimulus, and support on this project. I would also like to thank Lieutenant Colonel James Rutledge and Captain Roie Black for their ideas and comments.

Additionally, I would like to thank Bill McQuay who provided me with the opportunity and working experience by sponsoring the study. I would also like to thank Second Lieutenant Eric Grunden for his invaluable assistance and Ms. Cheryl Nicol for the excellent typing support in preparation of this document.

Finally, I would especially like to thank my wife, Laurie, and son, Matthew, for their understanding and patience during the course of this project. Without their encouragement the project would never have been completed.

Contents

List of Figures

List of Tables

Abstract

This study initiates the development of an interactive computer graphics system. This included the integration of off-the-shelf CORE standard graphics software for use on several computer graphics refresh and raster display devices. The system was implemented on a Digital Equipment Corporation VAX 11/780 mini-computer in standard FORTRAN 77. The system's requirements and top-level design specifications were generated following a study of graphics design concepts. The operational requirements of Precision Visuals' DI-3000 and GRAFMAKER software packages were interfaced with the system requirements to obtain the system level design specifications. The concepts of good interactive man-machine communication were included in the final design. This produced a reliable graphics system responsive to designers of bar, line, or pie graphs. The system was designed using structured analysis and design techniques which will allow a follow-on study to implement recommended modifications or enhancements from the supporting graphics software.

EZDRAW-AN INTERACTIVE COMPUTER GRAPHICS PROGRAM TO DESIGN BAR, LINE, OR PIE GRAPHS

I Introduction

Graphs are reasonably simple and functional mathematical structures that can effectively represent large amounts of data used for analytic study. By having a computer process the data and control the graphical display, a variety of quality graphs can be produced in a timely and economical manner.

EZDRAW, the interactive computer graphics system of this study, produces graphs that are concise, readily understood, and pleasing to the human eye. This is significant for civil/military commanders or project managers who must make judgments and decisions based on the results of those studies.

Background

The Electronic Warfare Division, Air Force Wright Aeronautical Laboratories (AFWAL) recognizes the need for a management computer graphics system. The design of the graph layout and production into 35 millimeter or viewgraph slides is presently conducted by Department of Defense art resources and takes approximately 10 - 14 days. Although the Electronic Warfare Division has computer graphics devices capable of producing the required graphs, higher priority real-world tasks and the non-availability of trained

"computer graphics programmers" prevent effective utilization of these graphical display assets.

Problem and Scope

The basic problem in this study is the design and development of the software needed to provide a responsive interactive environment for a novice user to produce typical management graphs. An existing computer graphics system employing several computer graphics display devices and "off-the-shelf" computer aided graphing routines is to be used.

Approach

The approach taken to implement EZDRAW consisted of five principal steps. The first step was to study and understand current computer graphics technology and methodology. Through this study, much insight was acquired in regard to this scientific discipline. Also some of the problems inherent in current practice were identified in this step.

The second step involved a comprehensive definition of the requirements and specifications for the system. Included in this step was the identification of those pertinent concepts providing effective man-machine interface. The results of this step provided a basis for the actual design and subsequent implementation of the system. It additionally highlighted some considerations for future system improvements. Chapter II discusses the requirements definition process.

Step three of the implementation consisted of a complete

system definition including the identification of data-structures, input and output formats, support hardware and software, and the development of a preliminary user's manual. Chapter III documents decisions and considerations incurred during this step.

The next step involved the software development process wherein the FORTRAN program and associated routines to create the graphics data-structures and permit user interaction with the graphical structures were written. The development and operation of the program and routines are described in Chapter IV.

The final step incorporated test and evaluation procedures for the system. These procedures and their results are discussed in Chapter V. From the results of the tests, evaluations, and system output quality, recommendations for changes and enhancements to the system were derived and are discussed in Chapter VI.

Appendices documenting the processes outlined above are provided and include background information on basic graphical design concepts, a data dictionary, structure charts, subroutine reference maps, and the EZDRAW User's Manual.

II Requirements and Specifications

In the software development process, the detailed definition of the problem is followed by the requirements definition. Generally, this phase forms the basis for subsequent developmental processes since the requirements definition assesses the needs that a system is to fulfill and provides limits for the system design and implementation.

User Requirements

The Electronic Warfare Division, AFWAL, prepares results of analytic studies for special classified and unclassified briefings and for input to the Commander's Report. Additionally, one-time application programs are written to prepare the graphical charts on available computer graphics devices when time permits. What was needed was the development of a computer graphics system that executed interactively with a novice user with little or no knowledge of graphics design, computer graphics, or computer programming.

While performing this interactive function, it was required that the resulting graphics system have the following characteristics:

1. produce various types of graphs,
2. support various sizes and styles of lettering,
3. run on the computer and use the graphics resources currently available to the sponsor,

4. support regeneration of the created graph for modification or reproduction at a later date,

5. be capable of using data from user specified files or from data entered interactively by the user, and

6. be easily used.

The types of graphical forms provided by EZDRAW include two dimensional linear graph plots with multiple plotting capabilities, bar graphs, and pie graphs. A discussion of basic graphics design concepts used to develop the specific elements of each graph type produced with EZDRAW is provided in Appendix A. The elements of each graph type are also described later in this chapter under the system specifications.

Several graphical text editor options for positioning text strings and using various type fonts, styles, and character heights are provided for lettering.

EZDRAW was designed to be run on the sponsor's Digital Equipment Corporation (DEC) VAX 11/780 mini-computer. The computer graphics hardware include the RAMTEK 6212, the TEKTRONIX 4016, and TEKTRONIX 4014 (emulated on the DEC VT100 console terminal). The DEC VT100 "graphics device" is the primary display used during graph design sessions. The system also uses several graphics output copiers.

At the outset of the system design, there were two sources of graphics software. First, there was the FORTRAN graphics subroutines provided with the TEKTRONIX display

devices (i.e. PLOT 10). There was also TEKLIB, a library of graphics routines designed by the sponsor. TEKLIB consists of PLOT 10 graphics supplemented with special CALCOMP and RAMTEK routines. The CALCOMP routines provided various styles of lettering for text strings while the RAMTEK routines provided the color capability.

In the fourth week of the research quarter, the sponsor acquired two commercial graphics software packages. The first package, DI-3000, consisted of routines that provided graphics generation support. The second package, GRAFMAKER, provided chart/graph drawing routines.

EZDRAW was redesigned to the specifications of these off-the-shelf graphics packages since they appeared to fulfill the requirements of the sponsor and the goals of the study.

The problem of subsequent regeneration and reproduction was solved by using two file schemes. First, the DI-3000 package includes a sequential picture library called the METAFILE to which graphical output can be directed. This picture library of the graph can then be processed at a later time by another element of DI-3000, the METAFILE Translator. This is a stand alone interactive program that allows the graph to be viewed and manipulated on different display devices but has no editing capability (Ref 1:10). Although the METAFILE is a powerful tool, it did not solve the specific requirement.

A second file system was needed to provide the editing and modification desired. The graphic data structure built and maintained by GRAFMAKER is written to and read from a secondary storage device (Ref 2:A-18) using an external file system which the user can access directly for storage of the current graph or retrieval of existing graphs. EZDRAW allows the user to designate new or existing files for graphical data storage. Graphs are assigned record numbers by the user when entered into the file. When regeneration or modification is required at a later date, the user can retrieve the graph by file name and record number and continue graphical processing.

Data input for the graphs to be designed was structured to allow input from either the terminal (for interactive communication) or from designated data files attached during the graph designing session.

Ease of use was fulfilled in several ways. First, by using off-the-shelf graphics software tools, the graphics routines are reliable and well documented. Ease in programming and portability are also enhanced.

Secondly, the interactive scheme of the system was designed to provide a friendly user interface. The concepts of good man-machine dialogue were implemented and are described in the User Interface Specifications below.

User Interface Specifications

Perhaps the single most important consideration in the design of a good interactive program is that the program must be designed with the user in mind (Ref 3:125). One aspect of good man-machine interface is that the system should manifest a clear "master-slave" relationship (Ref 4:314). Also, the interactive system must be forgiving and easily accessed by the user. It must be able to provide help to the user at any time and it must be important to the user. Finally, the system must not behave like a person (Ref 5:116).

These characteristics are summarized and the manner in which they were integrated into the system design of EZDRAW is given.

A "master-slave" relationship should exist and should be clear as to whether the user or the system is the master or slave. The user is generally the master when the user has full freedom of choice as to the next step of the interface dialogue. The system is the master when the user options are severely restricted. Preferably, the user should be the master in a man-machine interaction. However, when dialogue efficiency demands it, this relationship can be reversed (Ref 4:314). When this occurs, the user should at all times be able to regain complete control by means of escape or interrupting mechanisms (Ref 4:315).

With a novice user, it may be advantageous to have the system control the dialogue until user proficiency and

confidence is attained.

To facilitate design operations, a variety of user initiated options were identified that provide all required display manipulations and data definition capabilities. These options are displayed in menu format for convenient user reference. For improved understanding, each menu entry describes, as closely as possible, the function it performs.

The use of menu selected options to control the interactive scheme of EZDRAW was selected since menus are very effective and conducive to keyboard operated programs. The use of menus in this system is especially useful to the novice user since menus help the user understand the range of alternatives and protects against invalid selections (Ref 6: 174).

EZDRAW menus are kept short and are subdivided into several sub-menus. Not only does this cut down the amount of time the user must scan the menu for the desired option, but the span of control for the software module commanding the menu remains manageable (Ref 7:199). The actual levels of sub-menus and subroutines controlled are described more fully in Chapter IV.

In an interactive environment, the exact response of the user to a specific situation (a prompt or an error message) can not be pre-determined. Therefore, the use of a keyboard was selected for its simplicity and user familiarity with a standard typewriter keyboard. The required input is

emphasized with descriptive prompts, error messages, and menu options.

The system should be forgiving when the user makes mistakes. When a user makes a mistake on input, the system will respond with helpful instructions and explanations. This allows easier recovery from the error. This will also give the user confidence of not being able to damage the system with minor errors and that serious errors will be caught by the system (Ref 5:117).

Due to the portability of EZDRAW, user access is defined by the display device being used for graphics output. In all cases, the access routines are simple and outlined in detail in the EZDRAW User's Manual (Appendix H).

At every point in the session, the user should be able to get any help necessary to use the system well (Ref 4:315). A user's operating manual as well as a computerized "HELP" feature at all levels of EZDRAW are available in support of the system.

An important but often neglected aspect to good user-system interaction is that the system should be important to the user. A system is taken seriously by the users only if they believe it helps them accomplish their jobs in a meaningful manner (Ref 5:118).

Finally, the system should behave like a machine. While friendliness is necessary, the system should not try to be "chatty" or mimic the language of the user (Ref 5:118). The

user-system dialogue in EZDRAW is straightforward and functionally oriented rather than conversational.

EZDRAW takes into account the possibility that the user may be untrained in the design of graphs and charts aimed at effective visual communication. Therefore, the system has default values which can be used as a guide and starting point until familiarity is gained with the system. This also prevents the user from creating poorly designed graphs during initial sessions. The default values can be redefined by selection of parameters that are entered by the graph designer during the actual design phase.

System Specifications

When designing EZDRAW, it was necessary to determine the basic elements of the three types of graphs to be drawn. By carefully integrating the concepts of good graphics design with the graphical elements allowed by GRAFMAKER, the graph components described below were obtained.

EZDRAW allows the user to label each graph with a main title and two secondary titles. The titles may be centered by default or may be positioned anywhere throughout the graph by the user.

Only one vertical and horizontal axes per graph are allowed in EZDRAW. This is aimed at reducing graph complexity and insuring that the scales used for each curve in the graph are consistent. Each axis may have labelled major tick

marks. The attributes of the axes themselves are set by the user and include line width, intensity, line style, and color.

Axis labels are provided by the user and are centered horizontally and vertically by default. For better visual effect, the vertical axis label may be rotated.

Tick mark labels are positioned implicitly by user designation of the tick marks that are to be labelled and rotation of the tick mark label with respect to the axis.

Additional text strings called notes are allowed to be positioned anywhere on the graph. EZDRAW sets an implicit limit of three notes per graph.

Each line graph and bar graph can have up to three different dependent data sets represented. That is, there may be three dependent variables for a single independent variable. The independent axis of these graphs can also be selected by the designer to be either horizontal or vertical.

The graph designer may also define the attributes associated with the curves of a line graph and the bars of the bar graph. For curves, this includes the appearance of the line representing the curve and the marker for the data points along the curve. For bars, this includes the appearance of the line border defining the bar and the color and pattern of the interior of the bar. Additionally for bar graphs, the user specifies the width of the bar and the desired offset from a tick mark along the independent axis.

Pie graphs on the other hand may have up to ten segments

although four to five is considered the optimal number of segments. The user may reposition the pie, designate in which direction the segments are to be drawn (clockwise versus counterclockwise), redefine the radius of the pie, and specify a rotation angle for the pie.

The appearance of individual segments of the pie are defined by the user. This includes the line border defining the segment and the color and pattern of the interior of the segment.

The additional option of exploding segments from the pie is available for further emphasis of data.

Text strings (titles, axes labels, and notes) can be defined by the user. The designer may specify text quality, font, inter-character spacing, text size, justification, rotation, and color. These text attributes and the manner in which they are used are described in the EZDRAW User's Manual.

III System Definition

The resulting interactive graphics system takes advantage of prepackaged software tools and the existing AFWAL computer and computer graphics hardware configuration. The components of the system depicted in Figure 1 are:

1. the program EZDRAW,
2. GRAFMAKER,
3. DI-3000,
4. the host computer,
5. the graphics display devices, and
6. the graphics output copiers (not shown).

Program EZDRAW

EZDRAW is the program that provides the interactive environment for the graph designer. It is written in FORTRAN 77, to be compatible with most computer graphics software and hardware.

EZDRAW consists of many levels which are consistent with its menu driven scheme. Figure 2 shows the menus available at each level of EZDRAW. The lowest level of EZDRAW is that provided by DI-3000 and GRAFMAKER. These routines are accessed through FORTRAN subroutine calls with the appropriate parameters, provided interactively by the user or as a system default value, passed during the call.

At the Command Level, the highest level of EZDRAW, a

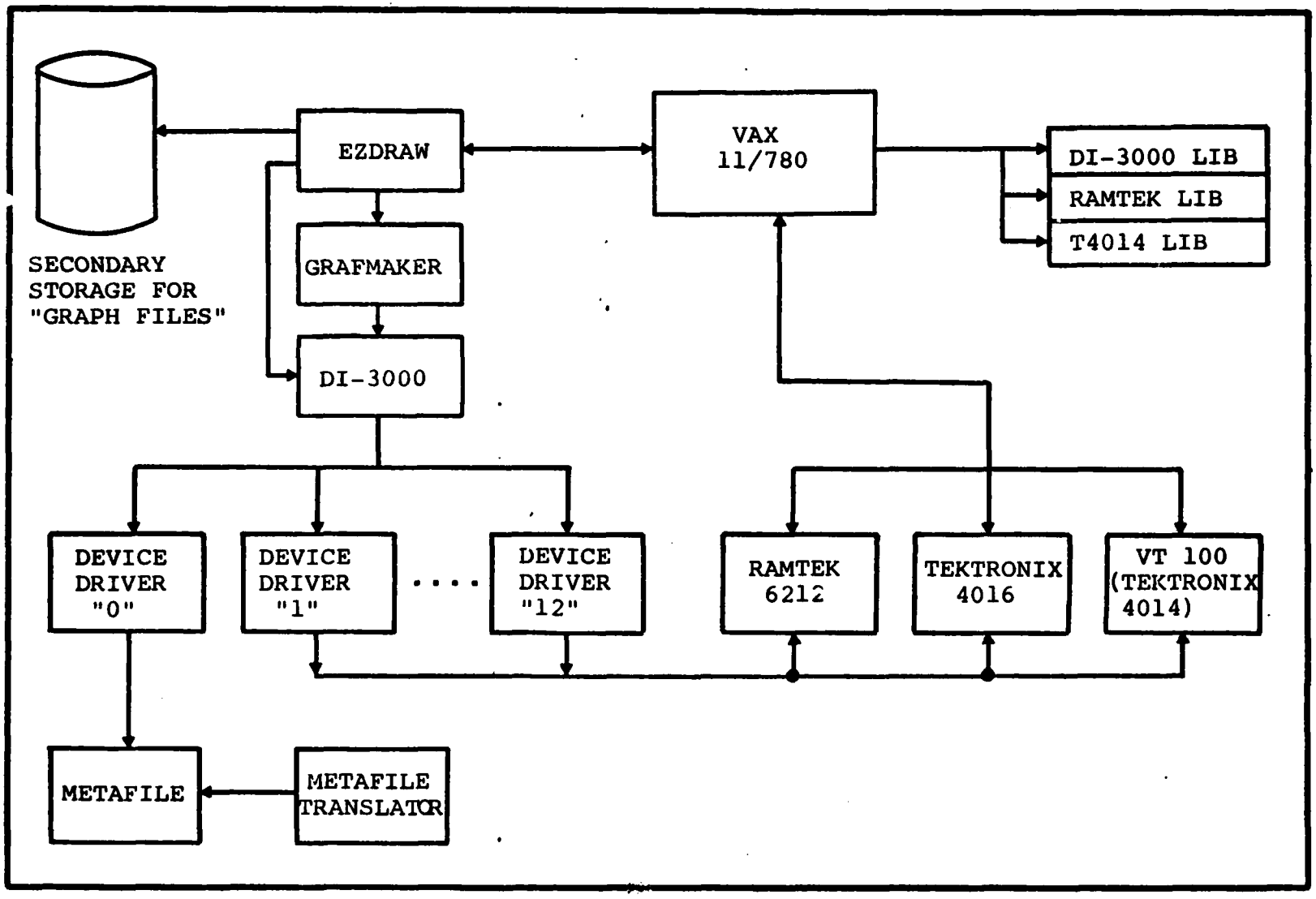


Figure 1. EZDRAW System Configuration

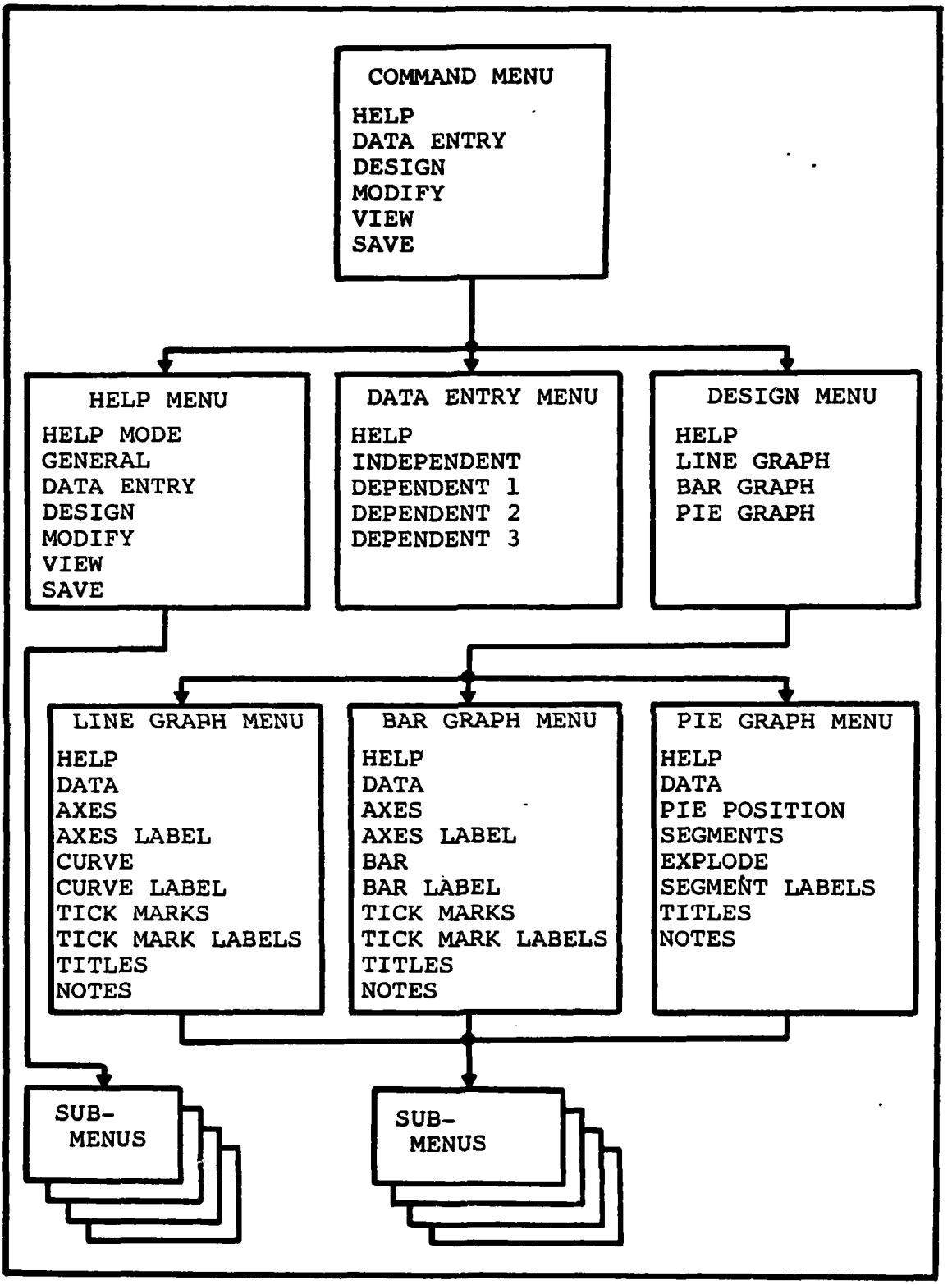


Figure 2. Representation of EZDRAW Levels By Menus

variety of operations can be performed on graphs. These operations include data entry, designing new graphs, modifying existing graphs, viewing existing graphs, and saving graphs. EZDRAW also has an on-line HELP facility that provides the user with useful information on graph design. The HELP feature and the graph operations are described in the following chapter.

The appearance of a graph at all stages of its design is also provided in EZDRAW. During the design or modification of a graph, the user is always "shown" the results of the user input. When the user wishes to work on a previously designed graph that was saved, the retrieved graph is again displayed to allow the user a visual inspection of the graph.

Extensive internal error checking is provided for all user required input, data entry, and file operations since interactive input is unpredictable and user dependent. Error messages respond to erroneous or invalid user input to menu and graph prompts. These messages indicate the type of error and then note the correct responses.

A comprehensive system of graph status flags is employed to prevent the user from producing conditions resulting in abnormal termination of the system. These fatal error conditions were identified in the testing of the GRAFMAKER picture data structure. The picture data structure created and maintained by GRAFMAKER is explained below while the test results and limits to GRAFMAKER are described in Chapter V.

Another feature of EZDRAW is that it protects against the user inadvertently overwriting the current graph before it has been saved. In this manner, system integrity is maintained and protected.

GRAFMAKER

GRAFMAKER is a set of device independent FORTRAN subroutines designed for use in preparing graphs and charts (Ref 2:1). These subroutines are called by EZDRAW and specify the appearance of the graph based on interactive input provided by the user.

GRAFMAKER is implemented in three sets of routines which are noted in Figure 3. The first set is for user interface and contains the routines invoked by EZDRAW. A listing of the GRAFMAKER routines called by EZDRAW is provided in Appendix F. The routines in this set process the GRAFMAKER picture specification commands, defining the data structure content for each picture. The second set is composed of those routines that manage and maintain the contents of the picture data structure. When the user indicates that the current picture be "drawn", the third set of routines interprets the data structure representation for the designated picture and calls DI-3000 routines to "draw" the picture on the display device (Ref 2:1).

A picture in GRAFMAKER terms consists of zero or more graphs and defines a region called the "frame space" or

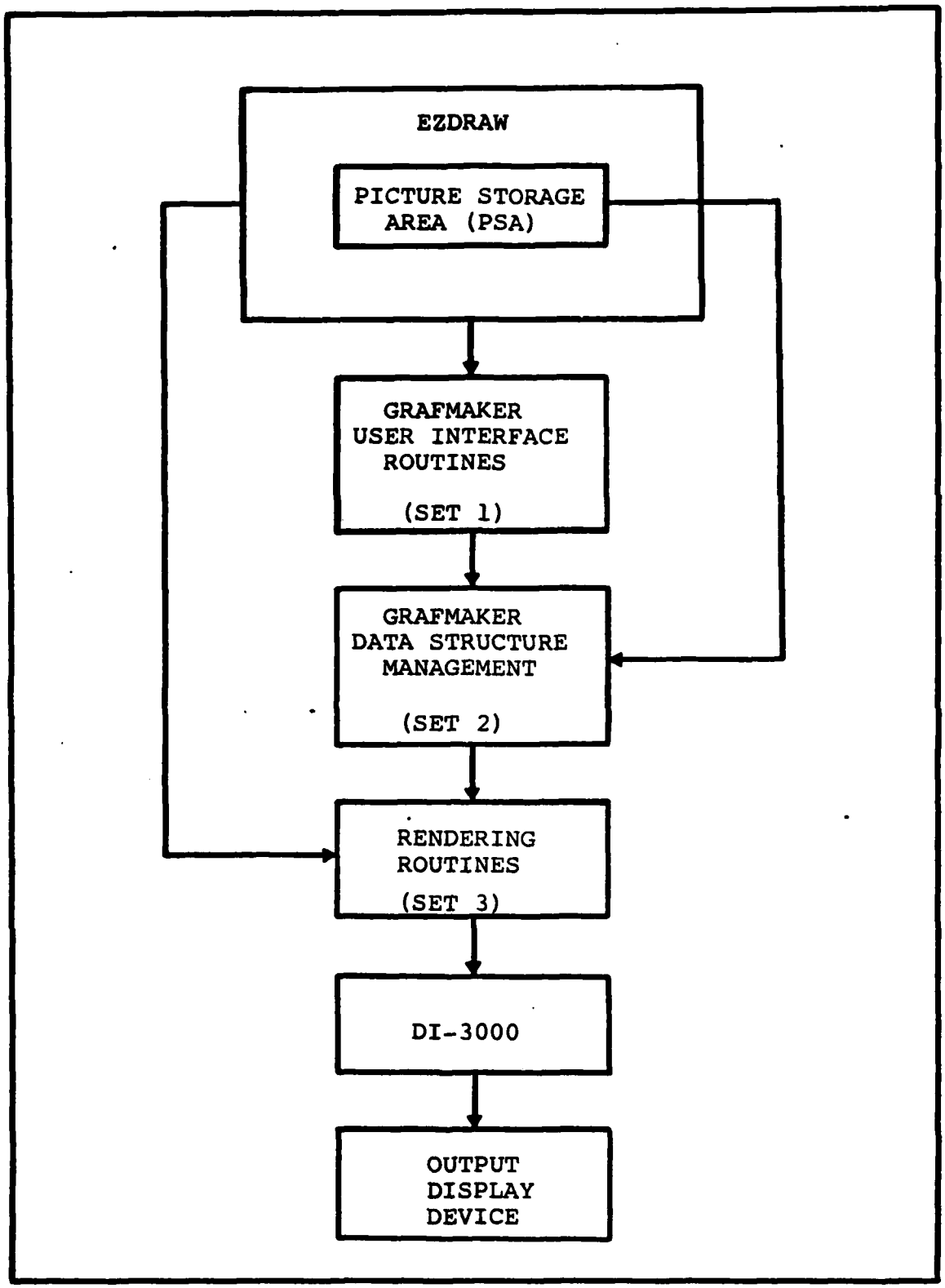


Figure 3. GRAFMAKER Design

"picture space". GRAFMAKER builds and maintains a data structure for each picture (Ref 2:2). This data structure is called the picture storage area and contains all information required to generate a picture consisting of one or more graphs. The size of the picture storage area determines the number of graphs that can be drawn in the picture (Ref 2:3).

In EZDRAW, simplicity is maintained by allowing only one graph to be designed and defined in the current picture space. This prevents user confusion and maximizes the data space for the graph. Another graph may be designed only after the GRAFMAKER picture data structure has been re-initialized.

Because GRAFMAKER maintains complete information in the picture storage area about a defined graph, this data structure can be written to disk from EZDRAW at any time. Later, the data structure can be read under user control and processing can continue from the point at which the picture was saved (Ref 2:A-18). This feature is supported by the use of a FORTRAN 77 external direct access file system.

DI-3000

DI-3000 is a packaged system of graphics software tools. It is implemented in 1966 American National Standards Institute FORTRAN as a library of FORTRAN callable subroutines. It has been designed and developed in accordance with the methodology and functional specifications of the 1979 CORE SYSTEM defined by the Graphics Standards Planning Committee

(GSPC) of ACM/SIGGRAPH (Ref 1:1). The DI-3000 subroutines are called by EZDRAW and GRAFMAKER subroutines.

DI-3000 is a modular software network which is depicted in Figure 4. In general, DI-3000 application programs call only the device-independent routines. These routines generate device-independent graphics commands allowing the program to build and interact with a virtual graphics device (Ref 1:1).

Device drivers are utilized for the device independence characteristics of DI-3000. A device driver is a set of subroutines which interpret device-independent commands and converts these commands into device-dependent instructions required to drive a specific graphics display (Ref 1:11).

The DI-3000 network manager coordinates the communication of messages between device-independent routines and the device drivers (Ref 1:2). Presently, the DI-3000 system implemented by the thesis sponsor can only designate one device per run of a DI-3000 application program. To change display devices, the program must be terminated and a link to the new device must be made. The exception is device "0" which is a METAFILE component.

A major element of DI-3000 is the METAFILE system. While graphics output is normally sent to a physical graphics display device, DI-3000 has the capability of storing the graphics output in a sequential picture file (the METAFILE), a physical graphics display device, or both concurrently.

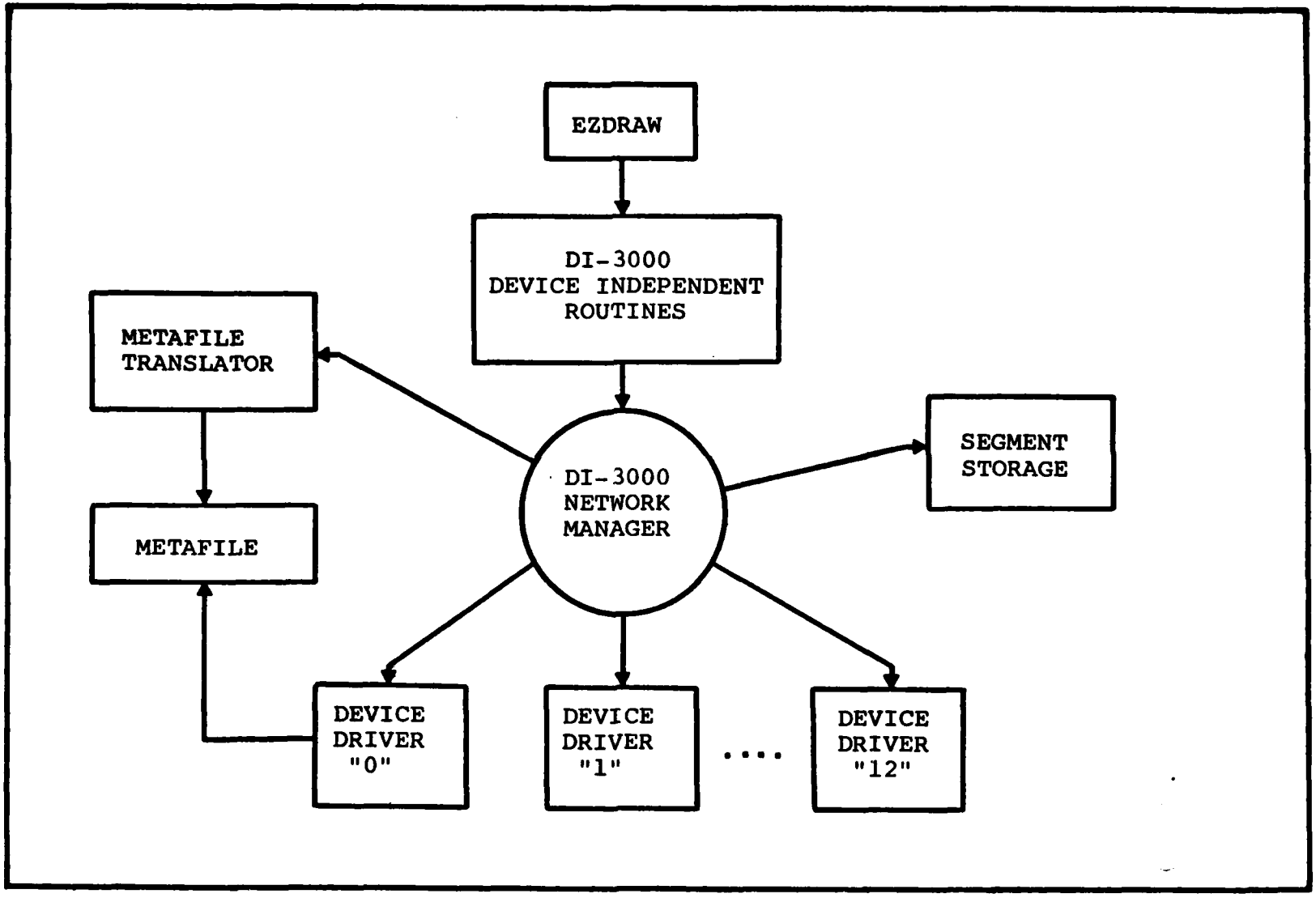


Figure 4. DI-3000 Configuration

This file of pictures can be written by activating device driver '0', the METAFILE Generator. Later, the METAFILE Translator can be used to process up to five different METAFILEs concurrently. In this manner, a capability to rescale and display more than one graph per picture frame (not available with EZDRAW) is achieved. Graphs are saved in the METAFILE during the EZDRAW "SAVE" operation at the user's option.

Host Computer

The thesis sponsor has a DEC VAX 11/780 serving as the host computer for the system. The VAX 11/780 is a virtual address extension of the PDP-11 computer family. The VAX 11/780 is a 32-bit machine and capable of a virtual memory of four billion bytes. The VAX/VMS Operating System is implemented and supports concurrent execution of multiusers, batch, and real-time application (Ref 8:1-2).

Display Devices

The computer graphics display devices (TEKTRONIX 4016 and RAMTEK 6212) are linked directly to the VAX 11/780. The computer and these display devices are located in a secure vault area within AFWAL. Because of the limited access to the computer vault, the capabilities of the TEKTRONIX 4014 have been emulated in several DEC VT100 terminals outside of the vault.

The DEC VT100 serves two purposes in this system. First, it is the means with which the user accesses and uses EZDRAW

while inputting the parameter specifications for the graphs.

Secondly, since the VT100 emulates a TEKTRONIX 4014, it serves as a low resolution graphics display device. Fortunately, the DI-3000 TEKTRONIX device driver is also capable of driving the VT100 in its emulation role. The rough display of the graph is presented to the user at the terminal.

The use of the VT100 restricted the interactive input of the user to the keyboard. Since there was not a device driver for the VT100, more responsive and natural means of graphics input such as cursor positioning or light pens were not available for the system.

In this configuration, color is not available with the output. However, this is overcome by the fact that DI-3000 and GRAFMAKER will compensate for parameters passed that are not available on the designated display device (Ref 1:3). Also, the VT100 has a much smaller screen which limits and reduces graph clarity. When the user is satisfied with the graph design, the actual physical device desired for graph output can be selected and enabled.

Graphics Output Copiers

An additional element of the EZDRAW graphics system is the hardware peripheral to the graphics display devices. These graphics output copiers are the Model 4007 Color Graphic Camera and the TEKTRONIX 4631 Hard Copy Unit.

The Model 4007 Color Graphic Camera is a color graphic

camera system designed for recording photographic hard copy from the output of raster scan computer graphics terminals. Full color, black and white, and color separation images can be recorded. Therefore, if EZDRAW is being run on the RAMTEK 6212, the graphics output can thus be recorded on a variety of film media including 8 in. by 10 in. Polaroid type instant print film, 4 in. by 5 in. film, 8 in. by 10 in. transparency film and 35 mm slide film (Ref 9:E).

When graphics output is being displayed on the TEKTRONIX 4016, permanent, high contrast copies of the output can be made with the TEKTRONIX 4631 Hard Copy Unit (Ref 10:1-1). Output from this unit can then be used to make viewgraphs for overhead projection.

IV Software Development

The actual design of the EZDRAW software was the culmination of the detailed definition of the requirements and specifications of the user and the system. It is in this phase of the software life cycle that the overall structure of the system takes shape.

In all phases of the design, a structured approach was used. Graphical flowcharts and structure charts were used extensively in the design of the EZDRAW system software. The structure charts used in the design of EZDRAW are included as Appendix B. Additional elements of software engineering such as modularity, programming style, documentation, and testing are encased in the EZDRAW design structure.

Levels of EZDRAW

In designing EZDRAW, a top down approach was used. The first step was to identify the levels of EZDRAW and implement the code for each level. The internal levels of EZDRAW are noted in Figure 5. These levels parallel those depicted in Figure 2, the "menu" representation of EZDRAW.

The Main Level of EZDRAW (Level 0) consisted of some basic preparatory operations to include initializing GRAFMAKER and DI-3000, initializing EZDRAW data arrays, establishing EZDRAW default values for the attributes of text and linear graph elements, and a graph element status flag array.

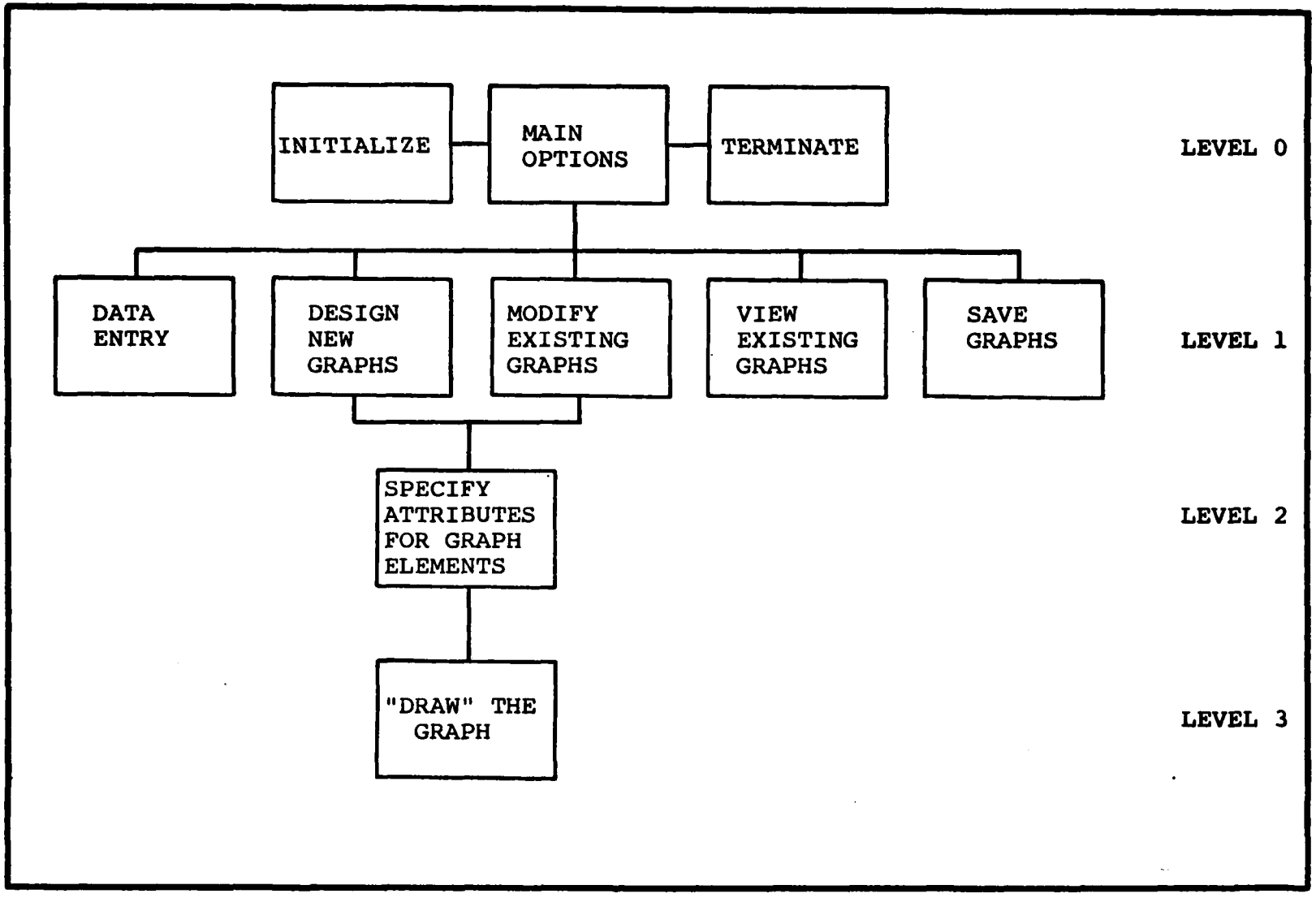


Figure 5. Levels of EZDRAW

After these preparatory operations, EZDRAW presents the main graph options to the user. This module also provides access to the lower levels of EZDRAW. Finally, the termination operations for EZDRAW are included in this level.

Level 1, the next lower level, is the EZDRAW Command Level. At this level, the user selects from the Command Menu the specific graph operation to be performed. These graph operation options were based on what a user could actually do to/with a graph:

1. Enter data for a graph,
2. Design the graph,
3. Modify the graph,
4. View the graph and make a copy of it, and
5. Save the graph.

The options of entering graph data, viewing graphs, and saving graphs are contained within the Command Level. On the other hand, designing and modifying graphs leads to lower EZDRAW levels. These Command Menu options are explained more fully in the Graph Operations section of this chapter.

After the user indicates what type of graph is to be designed or modified, the user selects from either the line, bar, or pie graph menu, which element of the respective graph will have its attributes defined and passed to GRAFMAKER. The specification of graph element attributes occurs in Level 2. It is at this Interface Level that EZDRAW interacts with GRAFMAKER. The interactive environment of EZDRAW is most

apparent at this level.

Level 3, is a pseudo-level in that there is no user interaction. At this Drawing Level, GRAFMAKER and DI-3000 chart and graphics subroutines are executing independent of EZDRAW.

EZDRAW's Modular Design

The upper level routines were written first with lower level module stubs used to insure correct operation of the upper level. The stub modules were then written as independent programs or linked subroutines. The interface of the lower level module to its caller was then completed. This involved the detailed definition of EZDRAW variables to be used in the lower level routines. These variables are listed and explained in the Data Dictionary, Appendix C.

Modularity is evident throughout EZDRAW. In large programs, such as EZDRAW, it is common practice to break down the program into smaller components. Each component is further reduced until specific functions or operations can be completed within that component (Ref 7:176). In this manner, the program itself is easier to design and maintain.

Another modular feature of EZDRAW is that the subroutines are grouped together by function and level. In this manner, all the subroutines involved with performing Level 0 functions were grouped together. Next, all the subroutines comprising the EZDRAW HELP Mode were associated. Another

group was the subroutines related to redefining the EZDRAW default values for the attributes of graph elements. Those subroutines allowing the user to specify the actual attributes for the data, text, and line graph elements were also consolidated. The final group consisted of EZDRAW "utility-type" subroutines (i.e. error messages, continuing from a pause, drawing graph and menu borders, etc). This informal grouping improved the overall organization of EZDRAW.

Whenever a large program is broken down into modules, the designer is concerned with the cohesion, coupling, fan-in, and fan-out properties of the individual modules (Ref 7: 187).

Cohesion is a relative measure of how the elements of a module relate to the accomplishment of a simple, identifiable task (Ref 7:191). The modules of EZDRAW have a relatively high degree of cohesion in that they are invariably functionally structured. For example, in Level 0, a subroutine is geared to initialize GRAFMAKER, DI-3000, data arrays, and EZDRAW default values. While these events (implemented as separate subroutines) are independent, they are related in time and give the calling routine its strong cohesive characteristic.

The measure of intermodule strength is called coupling (Ref 7:187). Loosely coupled or independent modules are desired since systems characterized by loosely coupled modules tend to reduce system testing and debugging (Ref 11:62).

The programmer and designer can debug a module without having to know anything about the contents or functions of other modules. One means of measuring modular coupling in a system is to look at module interfaces. The more complex the interface, the less independent the modules. Simple interface without excessive switches or flags are indicative of the module interface in EZDRAW.

Two other module traits are fan-in and fan-out. If a module is subordinate to several other modules, the number of calling modules is referred to as that module's fan-in. A relatively high fan-in is desired since the called module reduces repetitive code within the program (Ref 7:198).

On the other hand, the number of modules subordinate to a particular module is defined as that module's fan-out. A module should generally have a fan-out from five to nine (Ref 7:197). In EZDRAW, upper level modules tend to have low fan-outs. Mid-level modules have high fan-outs since they basically consist of the menu drivers and calls to GRAFMAKER subroutines. While modules at these levels violate the general rule, the subordinate modules have strong cohesion, loose coupling, and high fan-in.

The actual fan-in and fan-out of all EZDRAW modules are noted in Appendix D. The fan-in of DI-3000 and GRAFMAKER subroutines are noted in Appendix E and Appendix F respectively.

Graph Operations

A brief summary of the graph operations available as options from the Command Menu follows. A more detailed description of each operation is provided in the EZDRAW User's Manual.

Data Entry. The data entry option was elevated to the Command Level after it became apparent that this operation was the first step of creating all the graph types. Except for text elements, all other graph elements are embodied in or related to the representation of data. As such, these elements can not be specified without first entering the data.

EZDRAW allows the user to input data interactively or from an external sequential data file consisting of one record. Input data may be of type real or integer. EZDRAW then converts all data to real and places them in a general purpose data array. Based on whether the data set is for the independent variable or one of three possible dependent variables, EZDRAW then transfers data from the general purpose data array to a specific data set array.

When the user wishes to use the data in the EZDRAW data arrays (data may be used for more than one graph), the data set must be associated with the graph. This is accomplished by linking the data set to the graph by selecting the "link" option from the respective graph menu. Only the data set entered interactively or read from the data file is linked

to the graph. The size of the data set is expected to be much smaller than the size of the array holding the data set. EZDRAW allows a maximum of 100 data points to be entered in each of its data arrays to be used for graph data.

Design. When designing graphs, the user specifies the graph elements and their attributes by selecting the option from the graph menu. This accesses the actual element design routines that also pass the user specifications directly to GRAFMAKER.

For every graph element available in EZDRAW, the corresponding GRAFMAKER and DI-3000 parameter was defined as either a system standard or user specified. The user specified parameters required prompts.

The use of prompts for each parameter was selected to simplify user input. Requiring the user to specify the parameters as a list of arguments was not used since these lists tend to get lengthy and are error prone (the order could be permuted or an argument could be omitted) (Ref 11:94).

The prompts used were designed to be descriptive thereby assisting the user in entering a correct response. In all cases, error checking of the user input occurs to insure the response was correct (in type and range).

After the user has entered all the required values for the graph element, the graph is displayed providing the user immediate feedback to the design. The graph menu is then presented along with the graph allowing the user to select

another graph element to design.

Modify. Graph modification is very similar to graph design. In fact, the modification routine invokes the same graph menus and menu drivers to specify the graph elements. The main difference is that EZDRAW must initialize the picture data area for new graphs by graph type. Therefore, in Design, the user is working with a "new" picture data area while in Modify, the user is working with an "old" picture data area.

Another difference between Design and Modify is that in Design, the user indicates the graph type and then EZDRAW links the appropriate graph menu to the "new" picture data area. In Modify, an "old" picture data area is retrieved (graph retrieval is described in View below) and the proper menu and driver must be linked to the picture data area. This required the declaration of variable that served as an index to the type of graph that was saved.

View. The graph viewing module allows the user to expand the graph to better utilize the display device screen size and resolution. When the enlarged graph is displayed, the graphics output copier attached to the graphics device may be used to make permanent hard copies of the graph.

The user has the option of viewing the current graph (the graph whose pictorial data resides in the picture data area in main memory) or a previously designed and saved graph.

To retrieve a graph, the user specifies the file name and record number in which the graph was saved. The record containing the pictorial data is written to the picture data area in memory and the graph is displayed. The user indicates if this was the graph desired. If it is not, the process is repeated and continued until the correct graph is obtained.

Save. The graph saving module uses FORTRAN 77 direct access files to save the picture. The record in the file consists of the graph type index, the picture data area, and the array of graph element status flags.

The user specifies the file name and record number in which the graph will be saved. EZDRAW creates new files for the user as desired.

Additionally, the user has the option of saving the graph as a DI-3000 METAFILE. When this option is exercised, then EZDRAW enables the DI-3000 METAFILE Generator (device driver '0') and displays the graph to the output device. In this manner, the picture data is transferred to the METAFILE.

The option to save graphs is also presented to the user as a specific prompt before EZDRAW initializes the picture data area and/or terminates. The picture data area is initialized when the design mode is entered and when graphs are retrieved for viewing and modification. EZDRAW terminates when the user selects the exit option from the Command Menu.

Other Software Design Considerations

In the overall software design, consideration was also given to the effective use of the programming language, programming style, the user environment, documentation, testing, and maintenance.

Although FORTRAN 77 and VAX FORTRAN (V2.0) have many useful structures and functions, EZDRAW used mostly the fundamental structures and functions that form the elements of structured programming (Ref 11:2). The power of FORTRAN 77 was used only in those few situations that necessitated GRAFMAKER compatability.

For example, GRAFMAKER does not support the FORTRAN 77 character variable type. Therefore, the "read" of a FORTRAN 77 internal file was necessary to convert a character variable to a form acceptable to GRAFMAKER (Ref 12:166). In these cases, documentation is provided within the code.

Additionally, the VAX FORTRAN structures which were available to reduce the amount of "GO TO" statements were not used. This adherence to standards optimized the portability of EZDRAW to other computers and computer graphics systems using DI-3000 and GRAFMAKER.

The actual coding also had bearing in the development of EZDRAW. A clean, simple, and consistent style was the goal of the coding effort. Indentations and blank lines were used to offset related structures. Nested conditional statements were formatted to avoid becoming "lost" in the logical

nest. The style adopted was also aimed at simplifying debugging and assisting the maintenance programmer (Ref 13:197).

Since the interactive environment was critical to this study, the code also considered those elements previously discussed in Chapter II. In summary, prompts are descriptive and informative; user input is checked for type and range; error messages explain the condition(s) leading to the error; menus are short while their options are functionally descriptive; and instructions are stated simply in non-technical terms.

In support of maintenance (debugging and enhancement) programming, documentation is comprehensive. Documentation assists a maintenance programmer since the environment of the maintenance programmer is much different from the environment of the system designer (Ref 13:199). Therefore, in EZDRAW all the major decisions impacting in the system design are evident. Since future enhancement or changes for optimization is expected, intermodule and intramodule relationships are defined explicitly with subroutine header comments.

Comments are used liberally when the code was not self-explanatory or sufficiently clear. They also highlight the cases requiring special FORTRAN 77 structures (i.e. reading internal files, concatenating text string delimiters, integer division, etc.). Additionally, when EZDRAW calls a GRAFMAKER or DI-3000 subroutine, comments describing the subroutine called are provided along with a GRAFMAKER/DI-3000 User's

Manual page number reference.

EZDRAW was structured with all the features of what is considered good design techniques. This was evidenced by the small number of design and coding errors encountered during testing. The testing of EZDRAW is discussed in the next chapter.

V Test and Evaluation

The testing of EZDRAW comprised the second major effort of this study. The goals in testing EZDRAW were to determine if EZDRAW executed in the manner in which it was designed and if EZDRAW fulfilled the thesis sponsor's (user) requirements.

Two secondary but equally important goals in the test phase were to insure that EZDRAW fulfilled GRAFMAKER requirements and that GRAFMAKER provided the graphics capabilities with which it was advertised. The latter of these secondary goals was of primary interest since the graphics package was installed after the initial EZDRAW system design was complete and the actual capabilities of GRAFMAKER were unknown.

Structured Testing

Module testing, integration testing, and system testing are three distinct operations that must be conducted in the testing of software (Ref 14:200).

In module testing, module drivers are simulated with software stubs to insure that lower level components are entered and exited properly. Groups of software components are tested together in integration testing followed by the testing of the complete system (Ref 14:200).

The top down and modular design of EZDRAW was conducive for testing in this manner. Not only was EZDRAW tested as a

system, but it was tested at its various stages of development.

Test Plan. To accomplish the testing, a test plan was developed early in the study. The plan included testing of the EZDRAW main driver (Level 0), testing of the major modules (the graph operations of Level 1), testing the main driver with the major modules, and system testing. The system test included GRAFMAKER testing, graph testing, and user testing. Each test phase was followed by an assessment to determine if changes were necessary.

Test Procedure. Each of the graph operations (Data Entry, Design, Modify, View, and Save) were tested as independent programs prior to linking with the main driver. Design and Modify contained stubs for the actual graph element design routines when integrated.

System testing began when all of the graph element design routines were implemented. The initial goal was to determine if EZDRAW could "create" simple graphs. When this capability was achieved, then the graph was saved and then recalled for viewing. Once these preliminary graph operations were functional, the testing of actual graph design (in the design or modify mode) was conducted.

With system testing, a system is presented with representative data which the system is expected to encounter and process rather than random data (Ref 14:200). In this manner, EZDRAW used data from non-EZDRAW designed graphs that was

acquired from the thesis sponsor.

To fulfill the secondary objectives, a wider range of data was presented to GRAFMAKER through EZDRAW. It was necessary to discover some operational limits and requirements for GRAFMAKER and make the appropriate changes in EZDRAW. Unfortunately, the field testing of GRAFMAKER was not as complete as possible due to the short period with which it was available.

EZDRAW was tested by the thesis sponsor during the early stages of system testing. Some changes (i.e. renumbering the items within menus, rewording the menu entries, and restructuring the menu to "fit" on the screen of all display devices are some examples) were incorporated as required. EZDRAW has also already been included as an active file in the sponsor's library.

Additionally, several AFIT students used EZDRAW to obtain graphs for use in their independent studies.

The most thorough test of a system is one that will attempt to execute all possible control paths and every statement in the program at least once with a set of data (Ref 14:201). This was neither plausible or possible with EZDRAW due to the size of the program. However, these concepts were exercised during module and integration testing.

Evaluation

During system testing, EZDRAW correctly allowed the

graph elements to be designed indicating proper utilization of GRAFMAKER. The graph testing highlighted a sequence in which graphs should be designed for best results. This sequence is noted in the EZDRAW User's Manual.

The most significant result of system testing was discovering that once a graph element was designed, it could not be altered. Attempting to do so would result in a fatal error condition and termination of EZDRAW, GRAFMAKER and/or DI-3000 ensued. This condition is especially noteworthy since EZDRAW was designed and structured with this capability in mind.

Since EZDRAW was designed for this dynamic modification feature (the ability to alter a graph element whose attributes have been passed as parameters to GRAFMAKER and the picture data structure), it was necessary to determine why the terminal errors were occurring. Precision Visuals, the designers of GRAFMAKER, indicated that the GRAFMAKER picture data structure manager routines do not allow altering of the picture data structure (except for initialization).

One of the reasons EZDRAW was designed around GRAFMAKER was the fact that GRAFMAKER built and managed a picture data structure for each graph. Additionally, vendor documentation seemed to indicate that dynamic modification was possible (Ref 2:1,5). The desirability of GRAFMAKER's picture data structure and associated manager routines for use in an interactive scheme such as EZDRAW decreased considerably at

this stage.

What is peculiar is that GRAFMAKER will allow the user to dynamically change a graph in the following manner:

1. reposition text strings (except axis tick mark labels) anywhere throughout the graph as often as desired, and

2. redefine the attributes for the pie of a pie graph (i.e. the location of the center of the pie, the radius of the pie, the direction in which the pie is drawn, and the rotation of the pie).

This constitutes a dynamic modification of a previously designed graph element (since the picture data structure is altered); however, text strings and their attributes, data, appearance of data (curves, bars, or pie segments), axes, and tick marks can not be changed.

Since dynamic graph modification was a major requirement for EZDRAW, a possible solution is outlined in Appendix G.

In the testing of EZDRAW, very few design errors were discovered. Additionally, the number of actual coding errors was minimal. This is attributed to the techniques of structured design that were used throughout the development of EZDRAW.

The majority of the errors were related to using GRAFMAKER properly (i.e. insuring that the correct GRAFMAKER subroutines were called and sequenced properly within the EZDRAW

module; defining the coordinates for the viewing space to avoid distortion).

VI Conclusions and Recommendations

As noted in the Test and Evaluation of EZDRAW, it was impossible to conduct a test for every conceivable condition that may arise during an EZDRAW session. Therefore, it is difficult to say that EZDRAW will execute exactly as specified in all cases. However, because of the strict adherence to structured design principals throughout the development of EZDRAW, users of EZDRAW can reasonably expect to obtain consistently accurate and correct output.

Conclusions

In its present form, EZDRAW can be regarded as a good, basic interactive graph drawing system. An example of each graph type is provided in Figures 6, 7, and 8. These examples also show all the graph elements available on each type of graph.

The following statements are therefore made:

1. With the exception of dynamic graph modification, EZDRAW fulfills the requirements of the thesis sponsor.

2. The interactive environment provided by EZDRAW is friendly and responsive to the graph designer.

3. EZDRAW utilizes the GRAFMAKER package effectively to produce quality graphs and charts.

4. EZDRAW supports the portability of GRAFMAKER.

Despite these favorable characteristics, two major

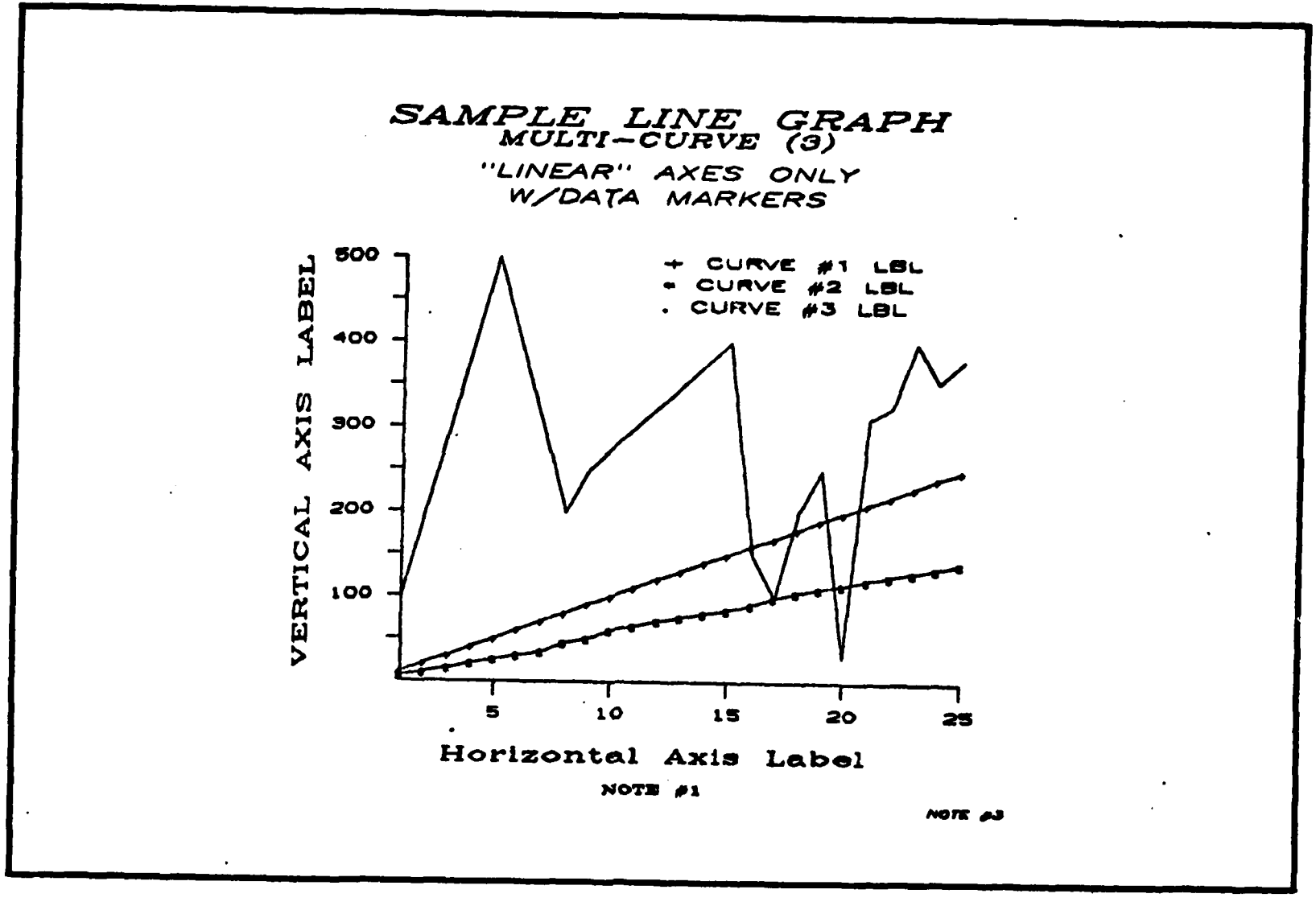


Figure 6. EZDRAW Line Graph

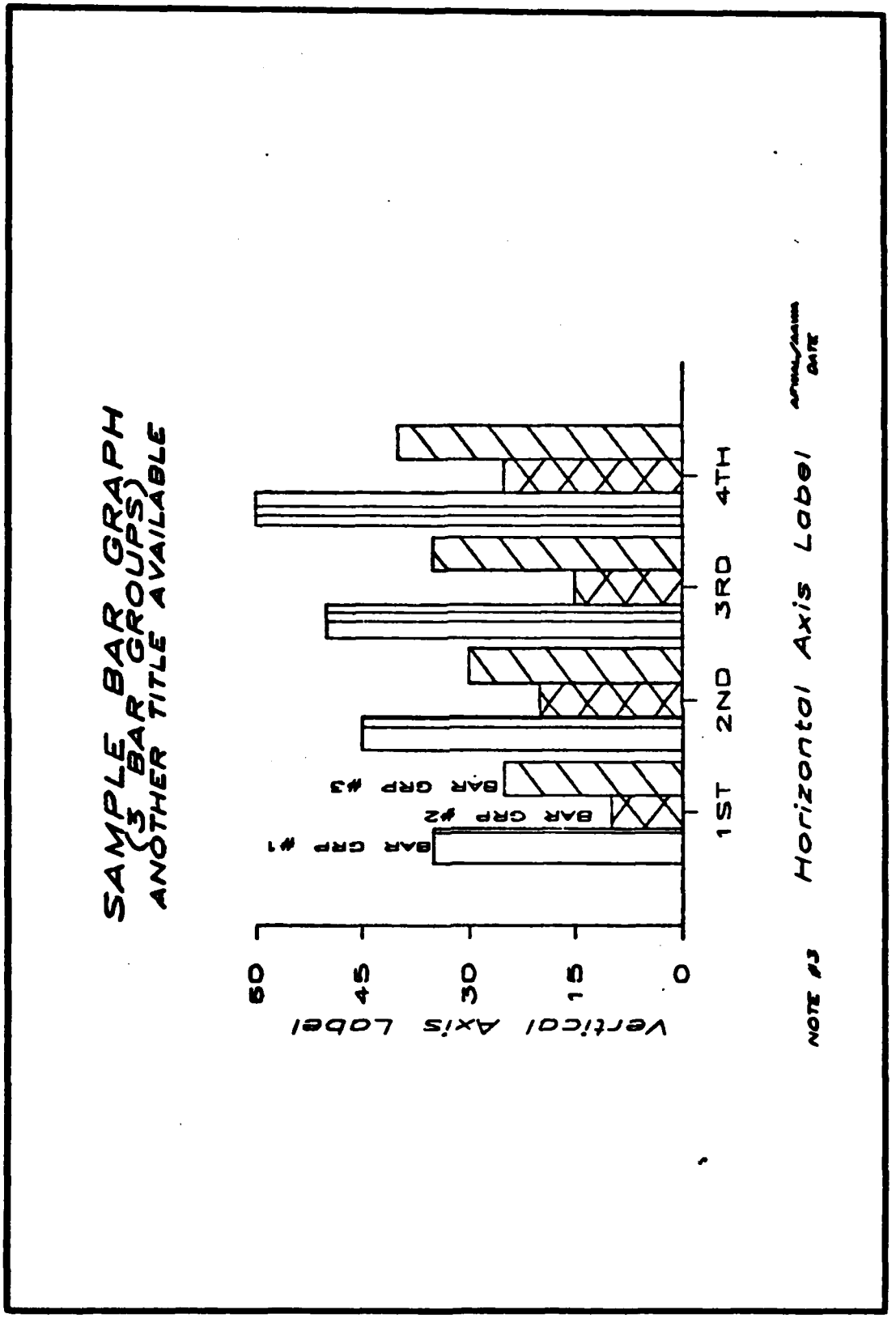


Figure 7. EZDRAW Bar Graph

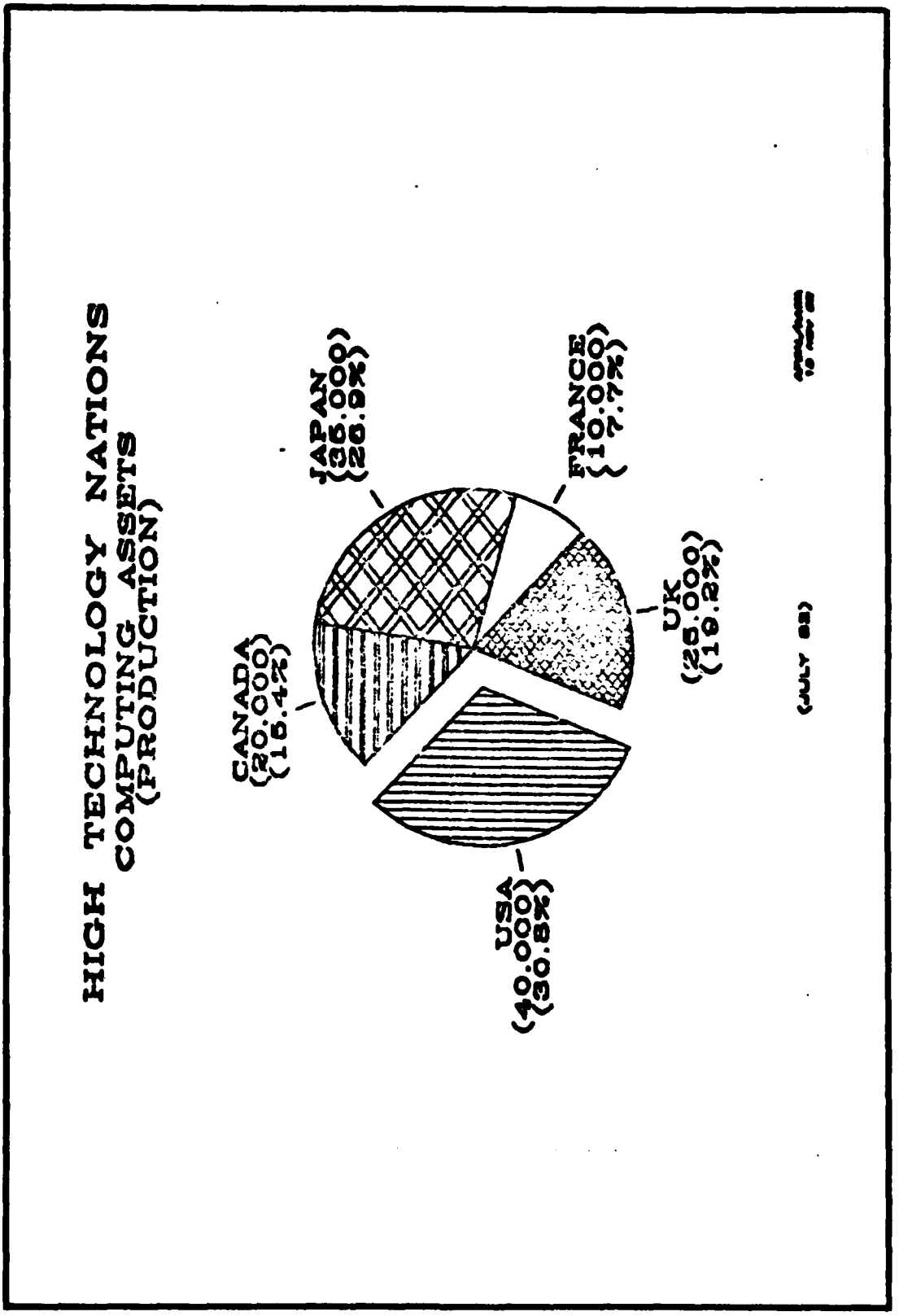


Figure 8. EZDRAW Pie Graph

drawbacks can still be ascribed to EZDRAW. The dynamic modification of graph elements is not allowed. The ability to change any element of a graph (once it has been specified) was a major goal of this study; but, as noted previously, EZDRAW's modification option is very restricted.

Secondly, user input is based on GRAFMAKER parameter requirements. The benefits gained from using off-the-shelf software was slightly diminished by having to model EZDRAW to the software package requirements. In EZDRAW, to allow the user the greatest flexibility in the graph design, the GRAFMAKER parameters were presented as user options rather than system standards. By doing this, the number of menus and the number of prompts may be considered excessive.

Recommendations

The following recommendations are aimed primarily at enhancing EZDRAW rather than redesigning the structure.

1. The dynamic modification feature should be implemented. This would allow EZDRAW to be more efficient and serve as a true graph designing system.

2. EZDRAW should be expanded to include a "text graph" mode allowing users to design graphs consisting only of text strings. This capability exists now, but the user must use the text strings allowed by the different types of graphs. The maximum number of text strings that can be designed on a graph is eleven (using the line or bar graph options).

3. A more efficient graph filing system should be incorporated so the user has information of the contents of the graph file during graph retrieval and save operations. This could be accomplished by creating a file directory as the first record in the file. The directory would contain graph identification data (provided by the user when the graph was saved). Upon retrieval, the user indicates whether the directory is to be displayed before entering the record number of the desired graph.

4. Additional GRAFMAKER options should be considered for implementation in EZDRAW. The most significant options would be those that allow the appearance of the graph to be changed without altering the picture data structure. These options are noted in the GRAFMAKER User's Manual as the set of "Change Routines."

5. The modules to implement logarithmic axes (base 10 and base e) for line graphs should be implemented. EZDRAW only allows linear axes although the user is presented with the option of specifying logarithmic axes. Selecting these options results in a message that these type of axes are future enhancements. The existing structure would easily accept this improvement since a data checking algorithm has already been coded into EZDRAW.

6. The modules to implement minor tick marks should be implemented while careful consideration should be given to implementing major and minor grid lines for line and bar

graphs. EZDRAW only allows major tick marks although the user is presented with the option of specifying these other possible graph elements. Selecting these options results in a message that these elements are future enhancements. The existing structure would accept these improvements readily.

7. The on-line HELP Mode should be expanded to possibly include a sample graph designing example.

Currently, EZDRAW can be used to design the three basic types of graphs used for management decision making. If the recommendations noted above are implemented in future versions of EZDRAW, the resulting graphics system would be a legitimate product for commercial application and distribution.

Bibliography

1. Precision Visuals. DI-3000 User's Guide. Boulder: Precision Visuals, Inc., 1982.

2. Precision Visuals. GRAFMAKER User's Guide. Boulder: Precision Visuals, Inc., 1982.

3. Schneiderman, Ben. "How to Design With the User in Mind," Datamation, 28: 125-126 (April 1982).

4. Dutch Graphics Interaction Standards Committee. "Some Issues in Interactive Programming Revisited," Computer & Graphics, 13: 314-319 (February 1980).

5. Moynihan, John A. "What Users Want," Datamation, 28: 116-118 (April 1982).

6. Newman, William M. and Robert F. Sproull. Principles of Interactive Computer Graphics (2d Ed). New York: McGraw-Hill Book Company, 1979.

7. Weinberg, Victor. Structured Analysis. New York: Yourdon Press, 1980.

8. Digital Equipment Corporation. VAX 11/780 Hardware Handbook, Maynard: Digital Press, 1979.

9. Matrix Instruments, Inc. Model 4007 Color Graphics Camera. Northvale: Matrix Instruments, Inc., 1981:

10. TEKTRONIX, Inc. 4631 Hard Copy Unit User's Manual. Beaverton: TEKTRONIX, Inc., 1982.

11. Kernighan, Brian W. and P. L. Plauger. The Elements of Programming Style. New York: McGraw-Hill Book Company, 1978.

12. Balfour, A. and D. H. Marwick. Programming in Standard FORTRAN 77. New York: North-Holland, Inc., 1979.

13. Clapp, Judith A. "Designing Software for Reliability," Computer Design, 20: 197-204 (SEP 81).

14. Zelkowitz, Marvin V. "Perspectives on Software Engineering," Computing Surveys, 10: 197-216 (JUN 78).

15. Myers, Ware. "Computer Graphics: A Two-Way Street," Computer, 13: 49-58 (July 1980).

16. Myers, Ware. "Computer Graphics: The Need for Graphics Design-Part One," Computer, 14: 86-91 (June 1981).

17. Marcus, Aaron. "Computer Assisted Chart Making From the Graphics Designer's Perspective," Computer & Graphics, 14: 247-253 (July 1980).

18. Szoka, Kathryn. "Graphics Primer for IEs: A Guide to Choosing the Right Chart Type," Industrial Engineering, 13: 74-79 (October 1981).

19. Myers, Ware. "Computer Graphics: The Need for Graphics Design-Part Two," Computer, 14: 82-88 (July 1981).

Appendix A

Graphics Design

Although computers are very useful in processing information quickly and accurately, the ideas and concepts the information represents must be effectively presented to be useful and understood. Computer graphics provides the means for successful visual communication when integrated with good graphics design concepts.

The basic reason for using graphics is that it conveys more information to the user. While the brain can perceive a picture in a glance, it must apprehend text or numerals in a slower, serial fashion. The brain must have a more efficient means of perceiving the meaning of a picture than thru decoding a string of letters or numbers (Ref 15:49). Graphics design assists the brain in this process.

Graphics Design Concepts

Graphics design is the conveying of visual abstractions from a source through a medium to humans. While the brain is the source, the medium is any form of representation the eye can apprehend (Ref 16:86).

Although the human visual system is a very sophisticated pattern recognition system, it does have clear limits to its perception of visual phenomena. Therefore, changes in color, line weight (a combination of line intensity and line width),

and other typical graph or chart-like information elements must be considered when conveying information visually.

The Swiss have taken an approach to visual organization of information that has been widely adopted (Ref Marcus:249). This Swiss approach to graphic design is suited to information display where complex relationships must be distinguished carefully and clearly. Some of the general visual characteristics that the Swiss approach encompasses are:

1. simple type styles,
2. simplified imagery,
3. open spaces,
4. design consistency, and
5. implied grid lines (Ref 17:249-250).

The traditional serif letter, a letter with a light line crossing or projecting from the end of a main stroke in a letter, has been rejected for a more simplified letter form. Variations in a family of simple stroke letters are acceptable and include bold, medium, condensed, or expanded.

Conforming with the choice of simplified letter type, the imagery of Swiss design emphasized reduced complexity, flat surfaces, and basic geometric forms.

The use of space void of text or illustrative material has also been adopted to add clarity and emphasis to the significant parts of the graphic material.

Consistency in design pertains primarily to not mixing letter types within the graphic material. The variations

noted in the family of letter types are accepted but should be limited to two or three.

Finally, a strong reliance on a grid of implied lines is evident. These lines organize and control the position of textual and illustrative elements. The grid limits the horizontal and vertical intervals within the graph and adds proportion, uniformity, and coherence to the fields of the graph (Ref 17:249-250).

Using these characteristics, studies have shown that the features described in TABLE I are useful in graphics design if properly restricted to reduce human perception error.

Graphs and Charts

Since it is the goal of this study to allow a novice user of the graphics system to create good graphs at the

TABLE I - Limits to Graphics Design Attributes

Usefulness of Various Graphics Features

Attribute	Maximum Number for Error Free Recognition
Color	6
Geometric Shape	10
Line Width	2
Line Type	5
Intensity	2

(Ref 17:249).

outset, the principles of graphics design must be incorporated in the graph drawing routines. These graphing principles as well as the design elements must be transparent to the user as much as possible.

Studies have also been conducted in regards to the effects of different types and styles of graphs and charts used in visual communication. Poorly designed charts overloaded with data may serve to retard rather than speed the viewer's acceptance of key points.

Selecting the correct chart to represent information involves two key decisions:

1. Choosing or creating a focal sentence. A focal sentence is one that summarizes what is to be shown;

2. Selecting a type of chart that illustrates that key sentence (Ref 18:74).

Before selecting the type of chart to support the data to be displayed, it is important to note that there are four basic types of relationships or patterns from which to choose in chart design. They are time series, parts of the whole, comparing several items, and relationships between two variables (Ref 18:78).

Time series, or the change over time of one or more dependent variables, is best displayed using the line graph or bar graph. The pie graph or bar graph should be employed when noting parts of a whole or percentages. The bar graph is appropriate when comparing several places or items.

Relationships between two variables are effectively depicted with line graphs or bar graphs (Ref 18:78).

There are some general guidelines to follow when employing any of these graphical modes. With line graphs, multiple curves are usually plotted. Hence, different line patterns/styles should be used for each curve. When comparing one curve against several others, it is best to single out that curve for emphasis, or else the graph may be difficult to understand. On the other hand, several graphs could be used together to compare one curve with each other individually with the same scale used in each of the plots. Multiple graphs per chart are also useful when larger volumes of data are being presented (Ref 18:75).

Use of graphs with double scales should be limited since they are often misread. Also, a scale change on either axis can alter interpretation. However, for analytical use, they can be effective for comparing relative changes. When scales coincide, a single set of grid lines could be used. In general, grid lines should be avoided.

Additionally, the curves should always be thicker than the grid lines and the axis to make the chart easier to read and there is no mistaking what is part of the curve. It is also preferred to place labels near the curves rather than using a legend (Ref 18:76).

A bar graph may be more effective than a line graph when there are few time periods. Bar graphs are especially

effective in showing large changes from one period to the next. The judicious use of reference lines and shading enhance the effect of bar graphs. The bars of the graph should always be wider than the space between them (Ref 18: 76).

Pie graphs are useful for comparing parts of the whole. Exploded pie graphs are especially useful because it focuses the eye on one segment of the graph. This can also be accomplished through shading of the segments. The number of segments to be used in a pie chart should be limited to four or five (Ref 18:77).

Other guidelines pertaining to all graphs and charts in general include always writing the axis labels horizontally (when read from the bottom and/or right of the graph) and making the axis numbers or tick mark labels large enough to be read easily. The scales should also make interpolation easy (tick marks should be added if necessary). Shade patterns should be ordered and simple (Ref 18:79).

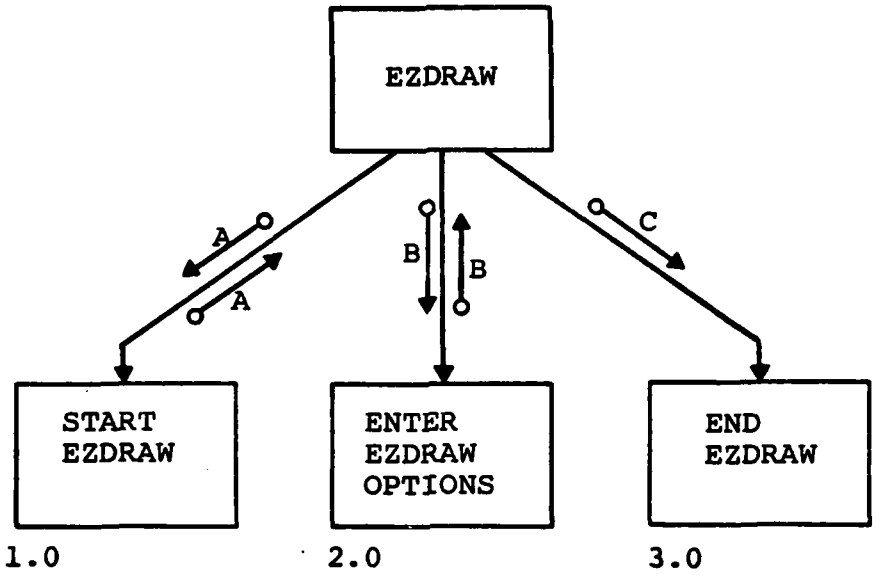
Symbols, mnemonics, abbreviations, and unfamiliar vocabulary should be avoided. Standard terminology if available is preferred. Too many data points plotted is confusing and lines should be fitted to the points. Fine lines as well as tiny grid lines imply precision. Therefore, line width and grid size should be consistent with the precision of the data (Ref 19:87). In all cases, a border should be used since it adds a sense of completeness and structure to the

elements enclosed within the boundary (Ref 17:251).

In the graph drawing routines of EZDRAW, the concepts of good graphics design have been implemented. With the interactive user-system dialogue, many elements of the graph being created are specified by the user. In some cases, limits and system standards have been imposed to ensure adherence to the visual syntax of graphical communication.

Appendix B

EZDRAW Structure Charts



A - PSA,PSIZE,SIZE,ASPECT,DEVNUM,GTYPE,FLAGS,
All data arrays, all line attributes, and
all text attributes

B - PSA,PSIZE,RCDLEN,SIZE,ASPECT,GTYPE,FLAGS,NUMSEG,
All data arrays, all line attributes, and all
text attributes

C - PSA,PSIZE,RCDLEN,ASPECT,GTYPE,FLAGS,NUMSEG

Figure 9. Main Module

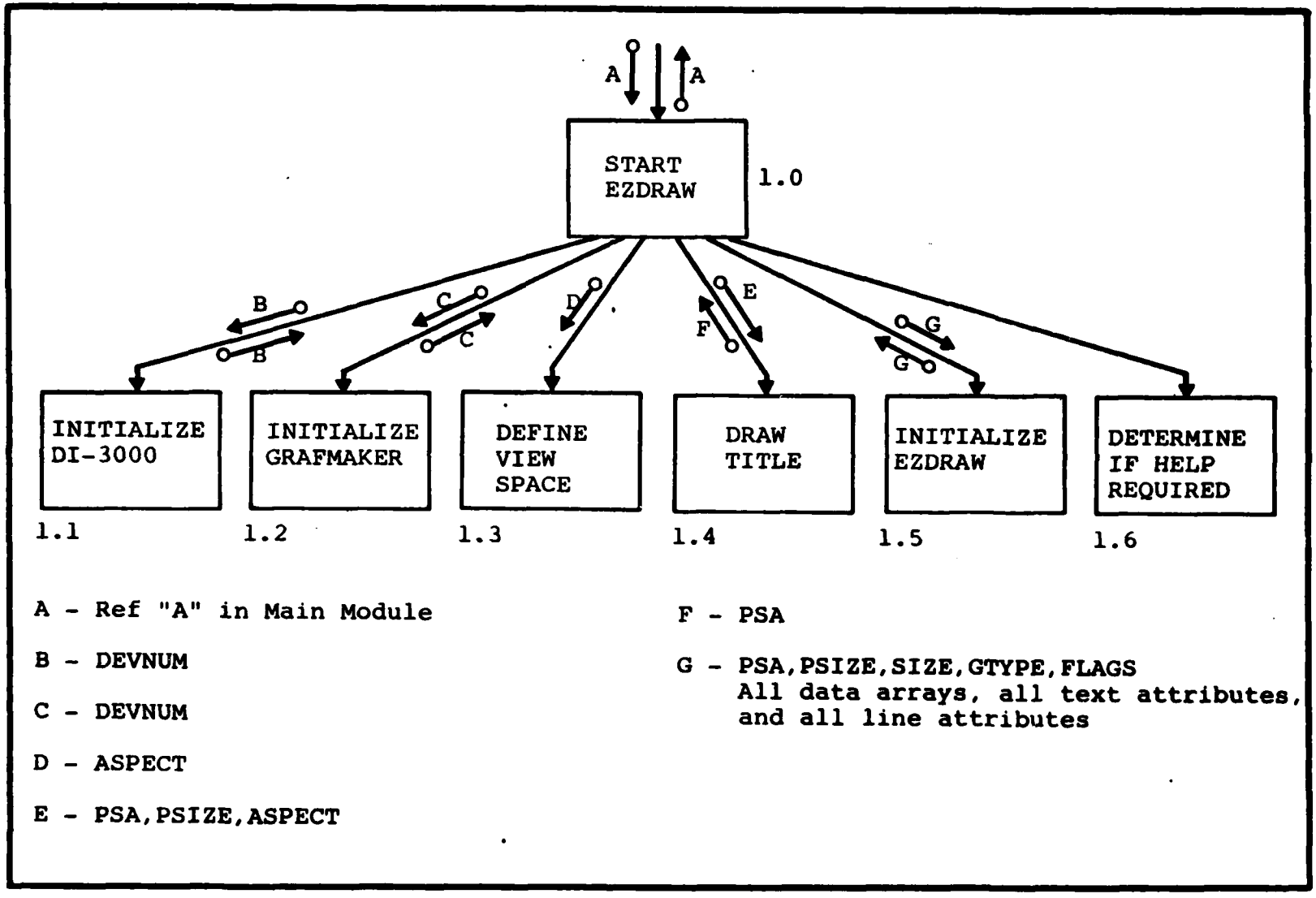


Figure 10. Starting EZDRAW

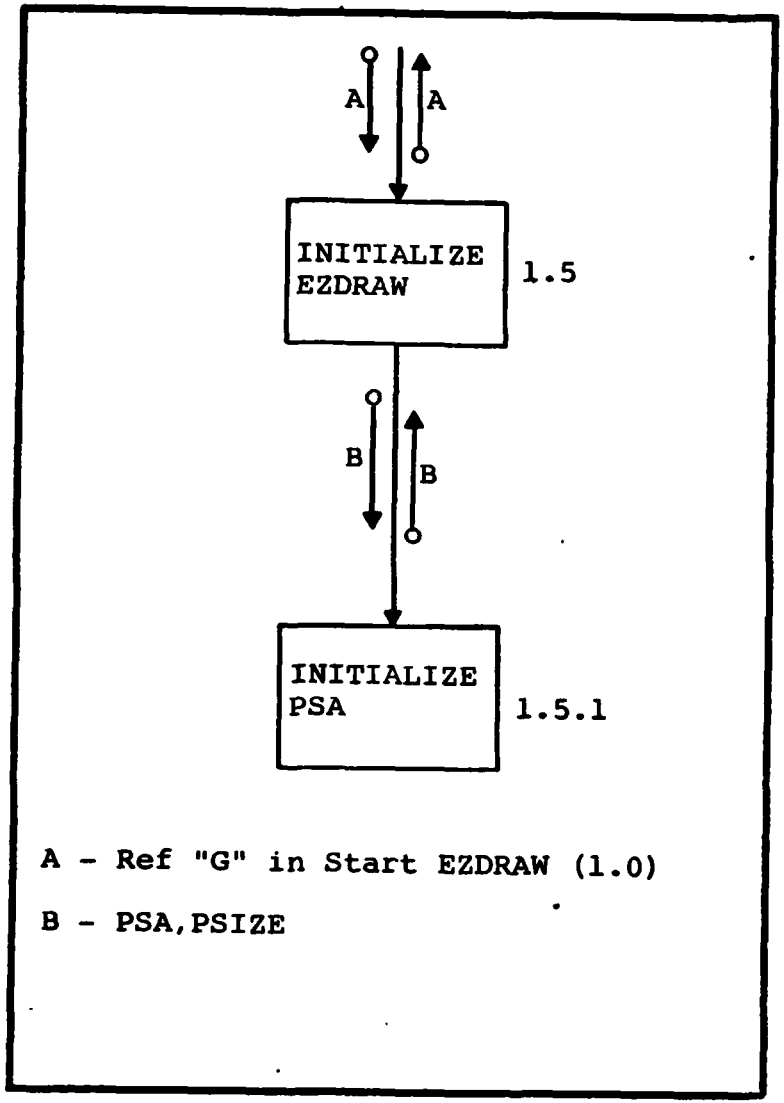


Figure 11. Initializing EZDRAW

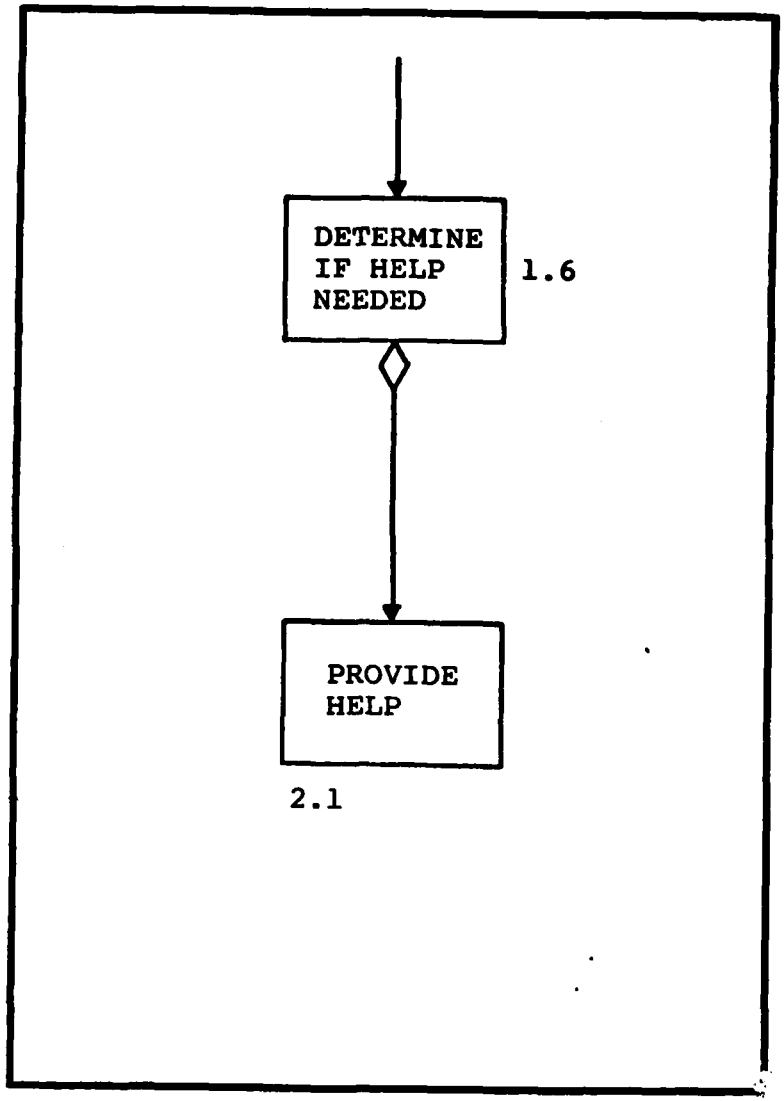


Figure 12. Determining if HELP Needed

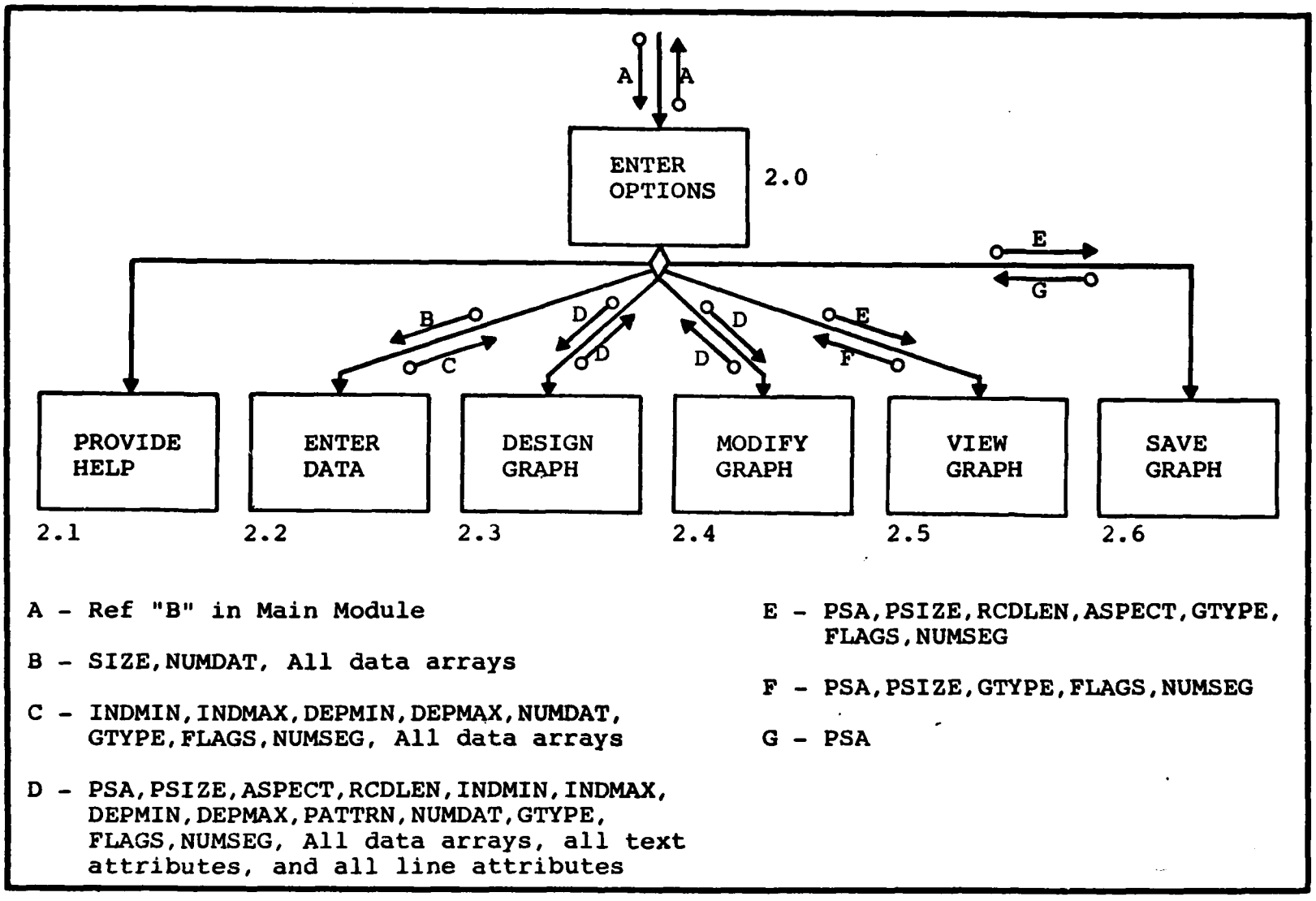


Figure 13. Entry of Main EZDRAW Options

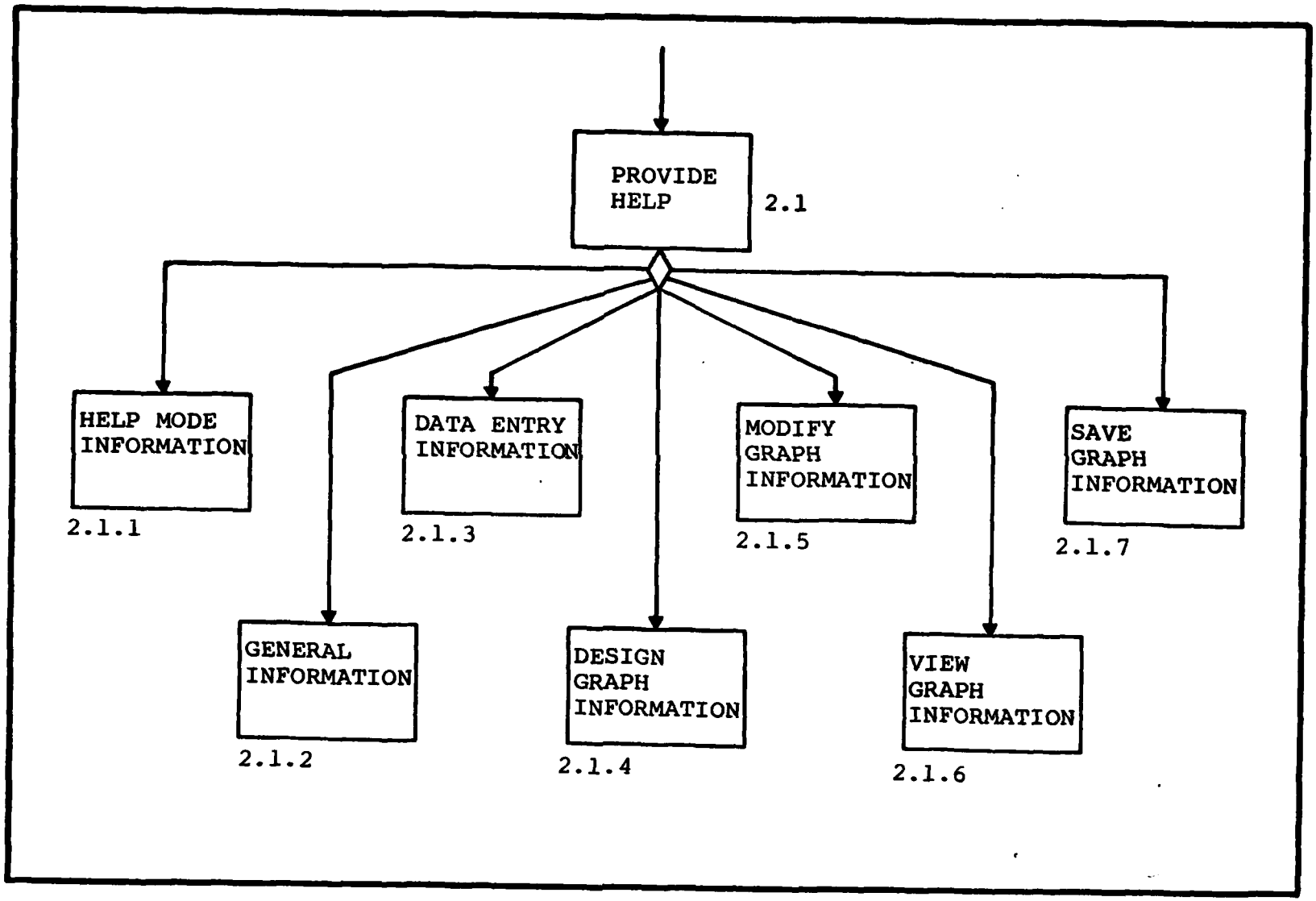


Figure 14. Providing the EZDRAW HELP Mode

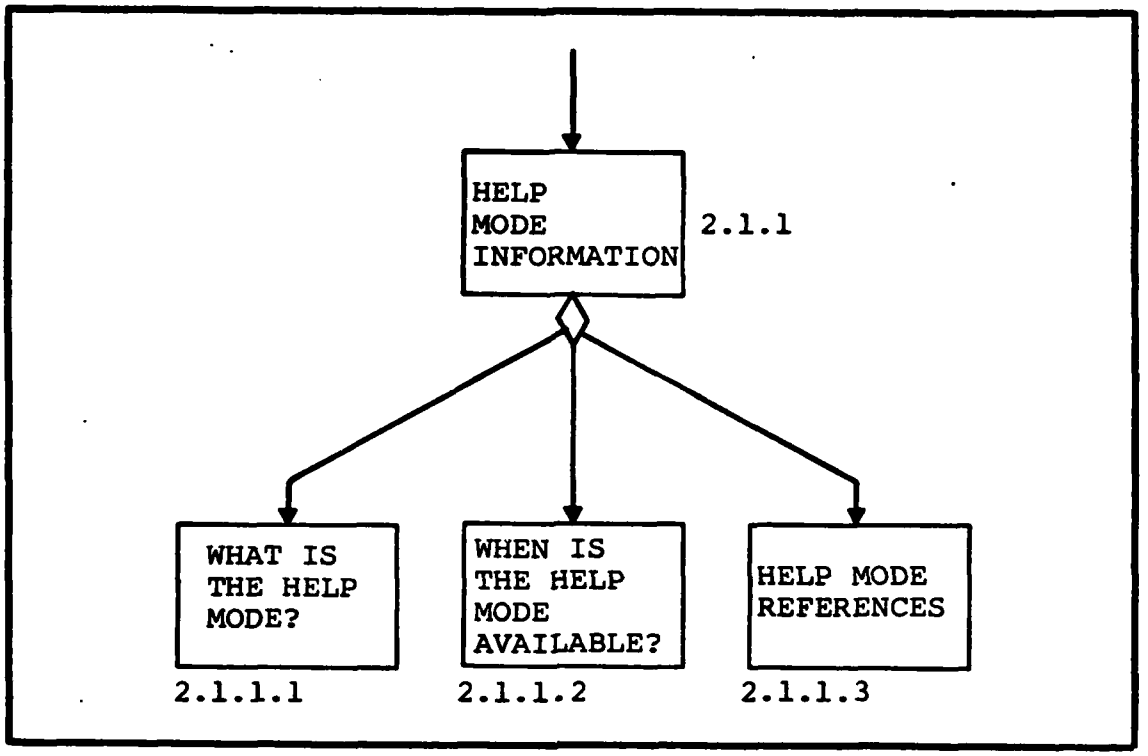


Figure 15. Providing Information on the HELP Mode

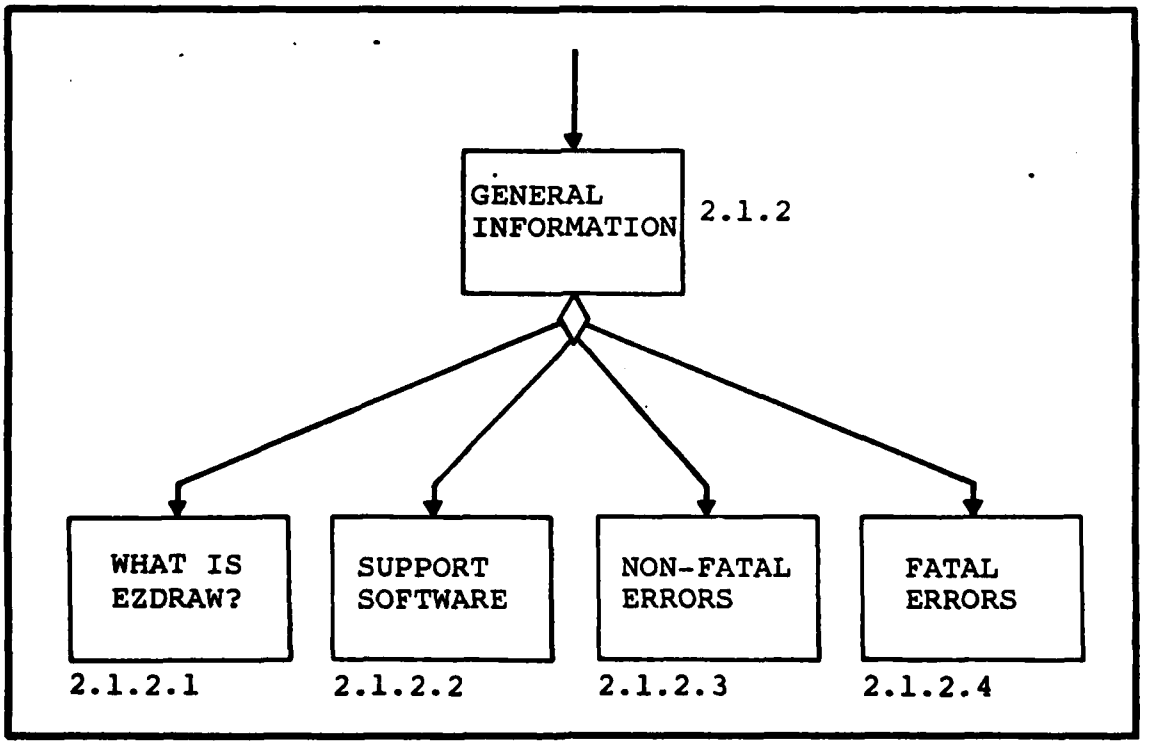


Figure 16. Providing General Information About EZDRAW

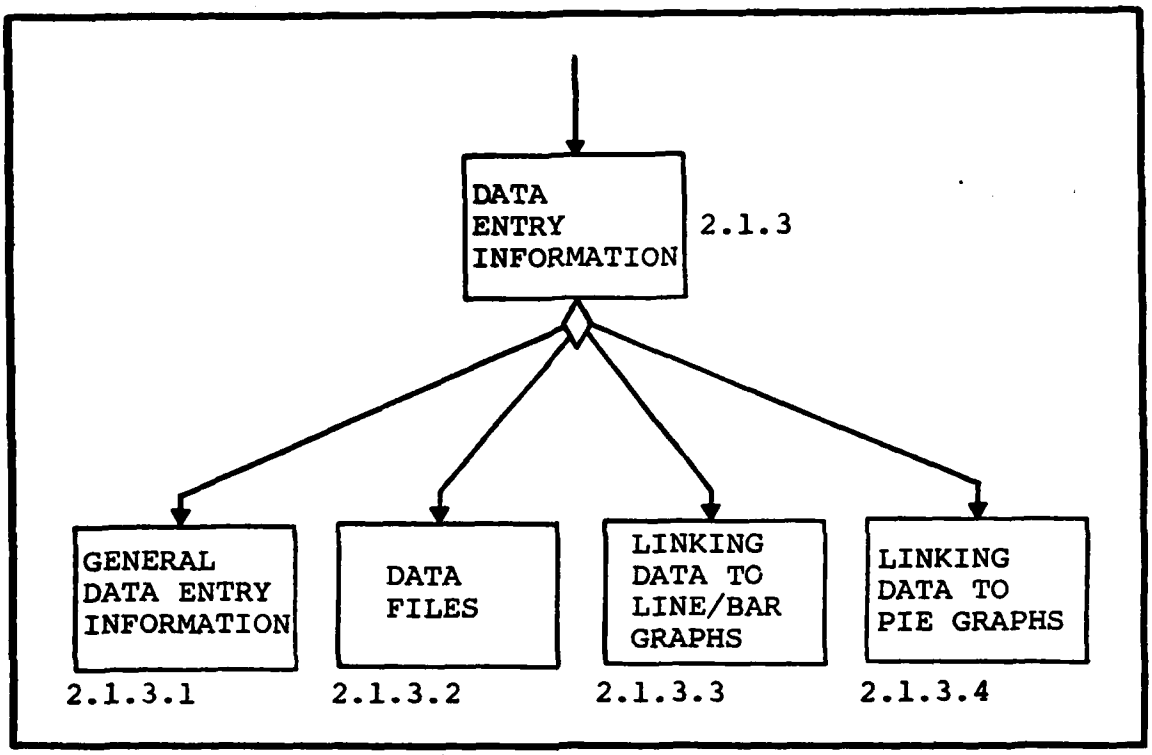


Figure 17. Providing Information About Entering Data

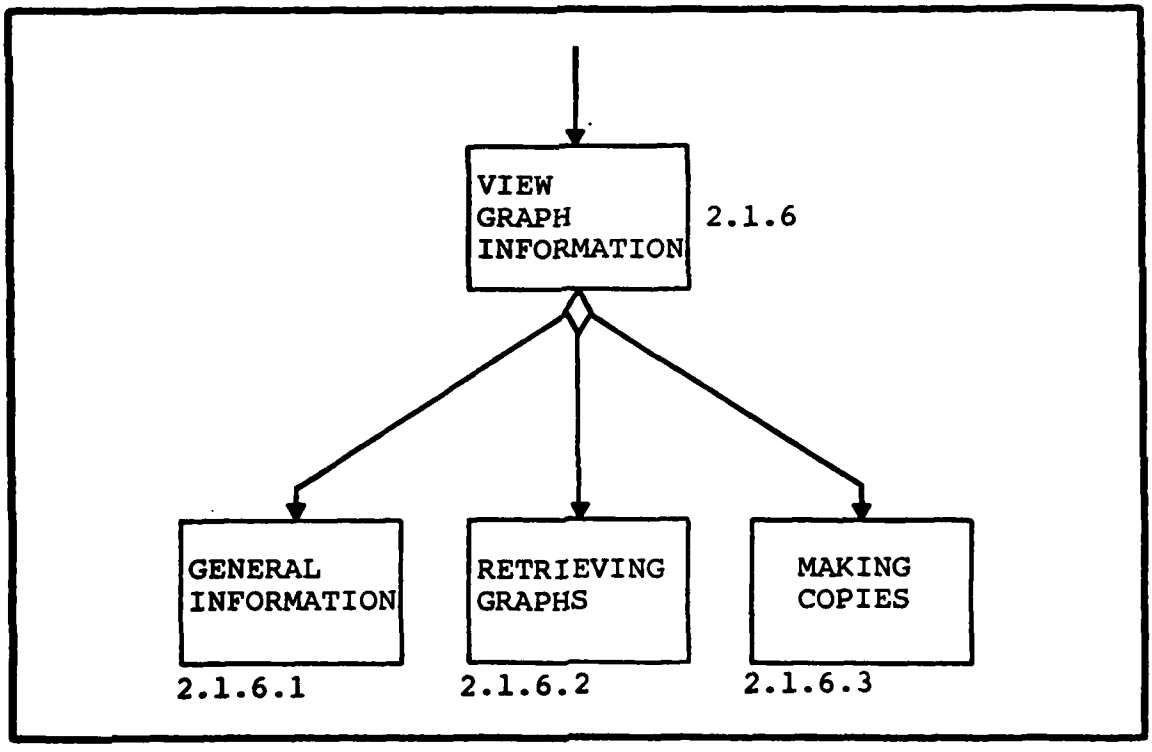


Figure 18. Providing Information About Viewing Graphs

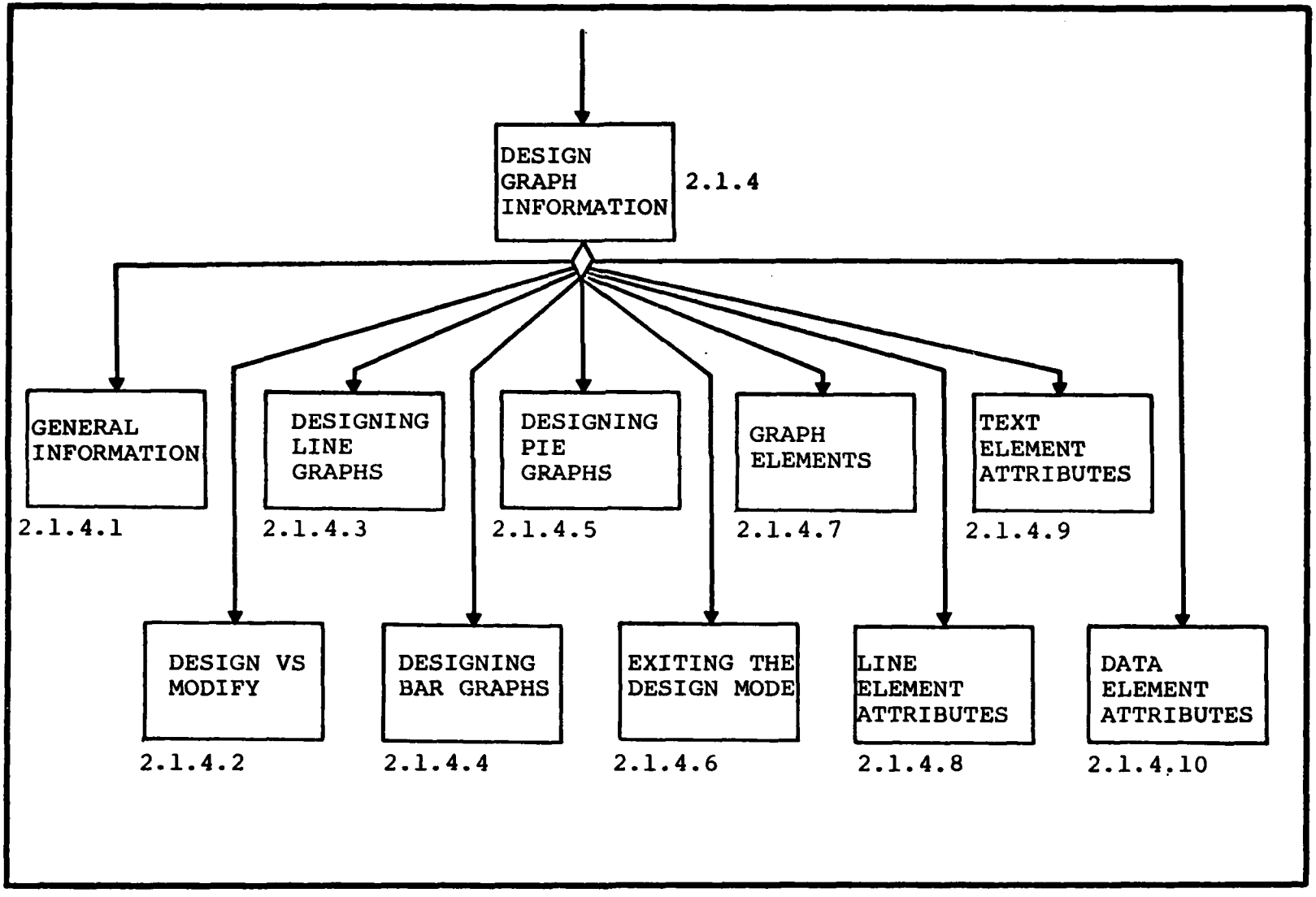


Figure 19. Providing Information About Designing Graphs

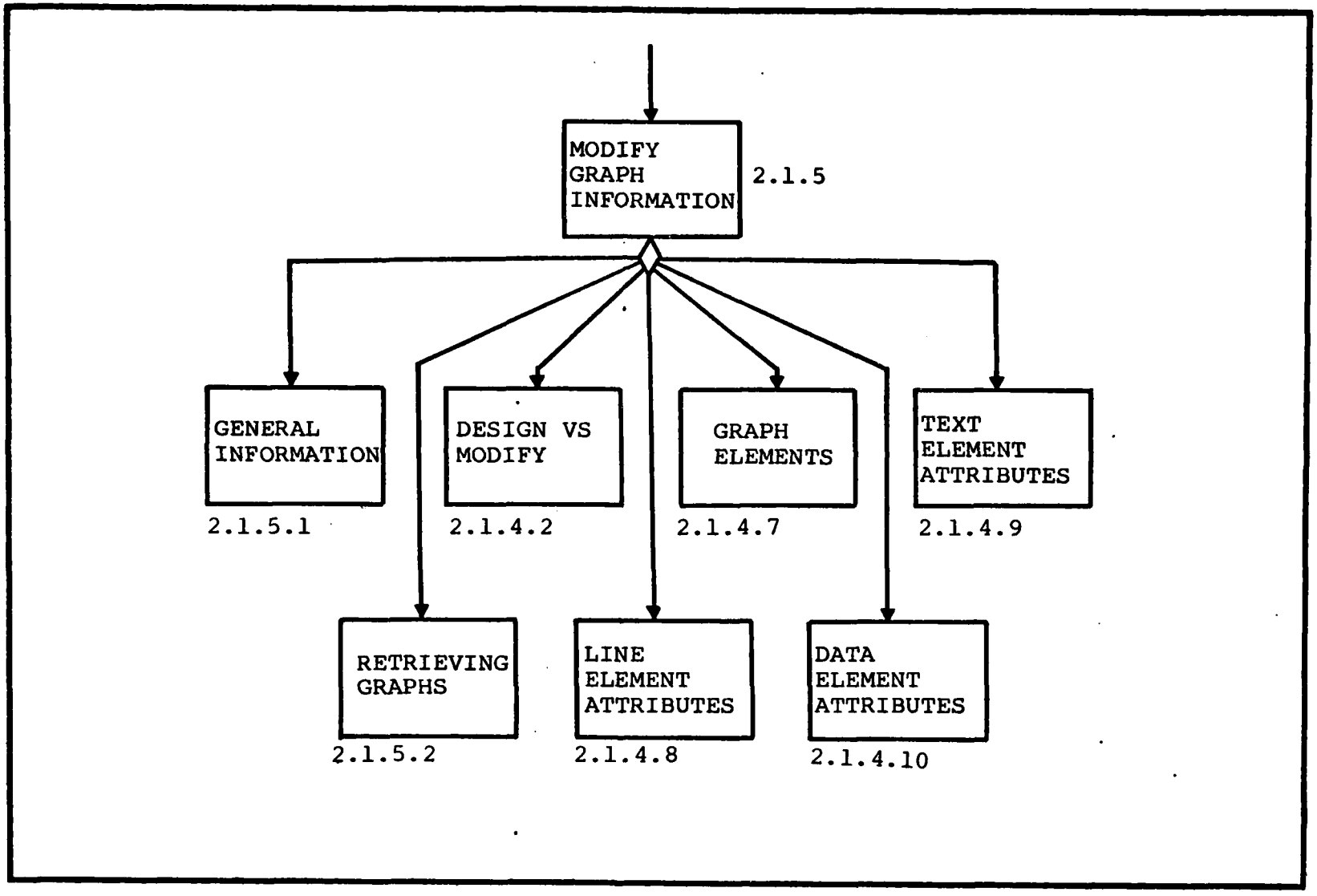


Figure 20. Providing Information About Modifying Graphs

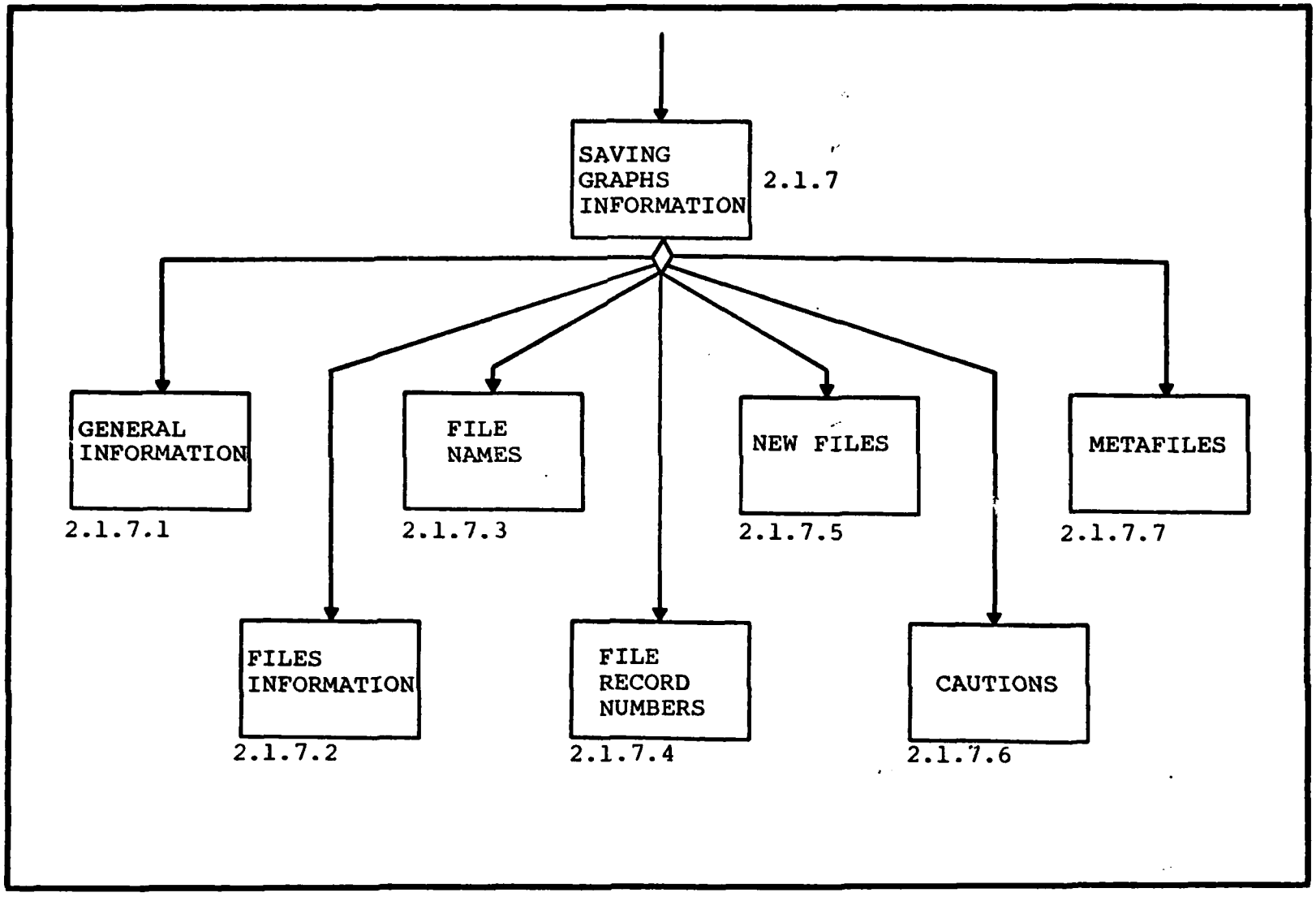


Figure 21. Providing Information About Saving Graphs

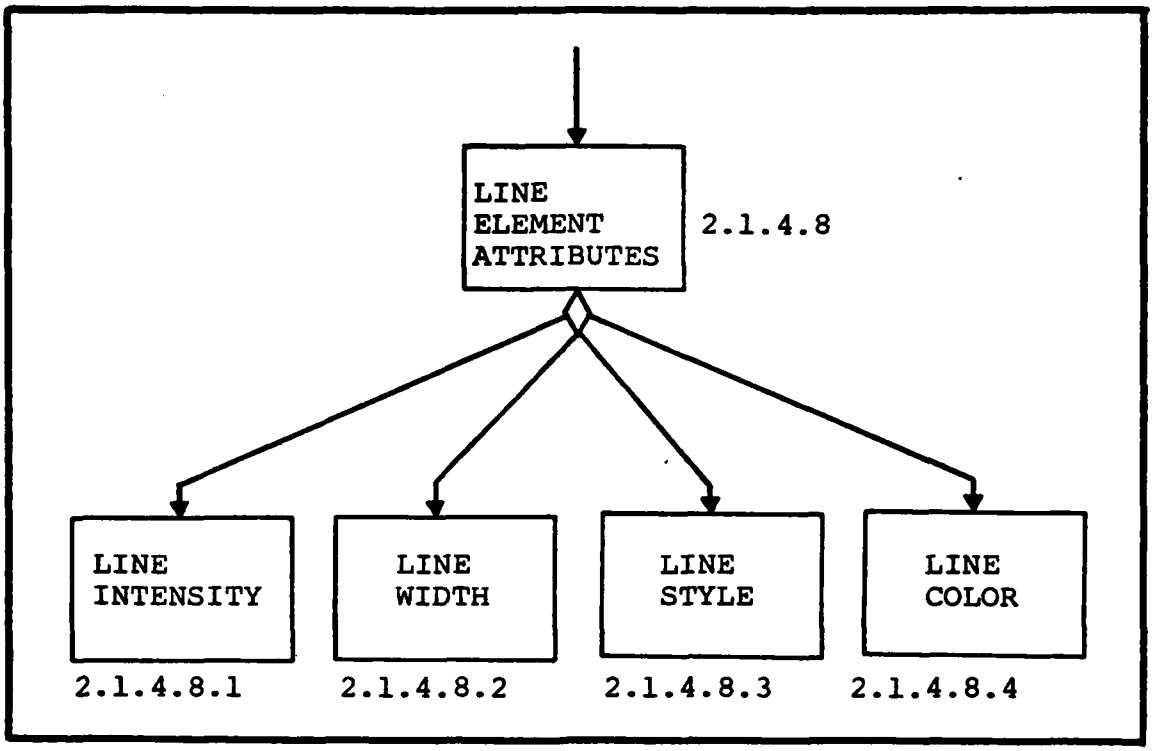


Figure 22. Attributes of Linear Graph Elements

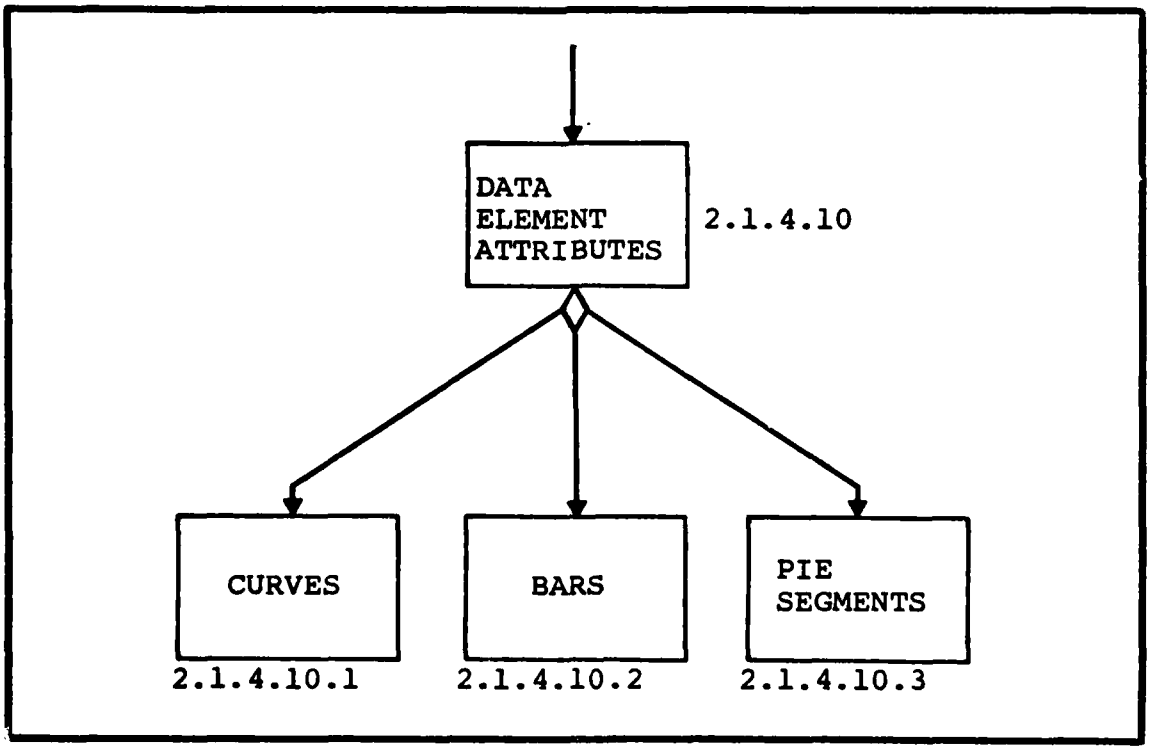


Figure 23. Attributes of Data Elements

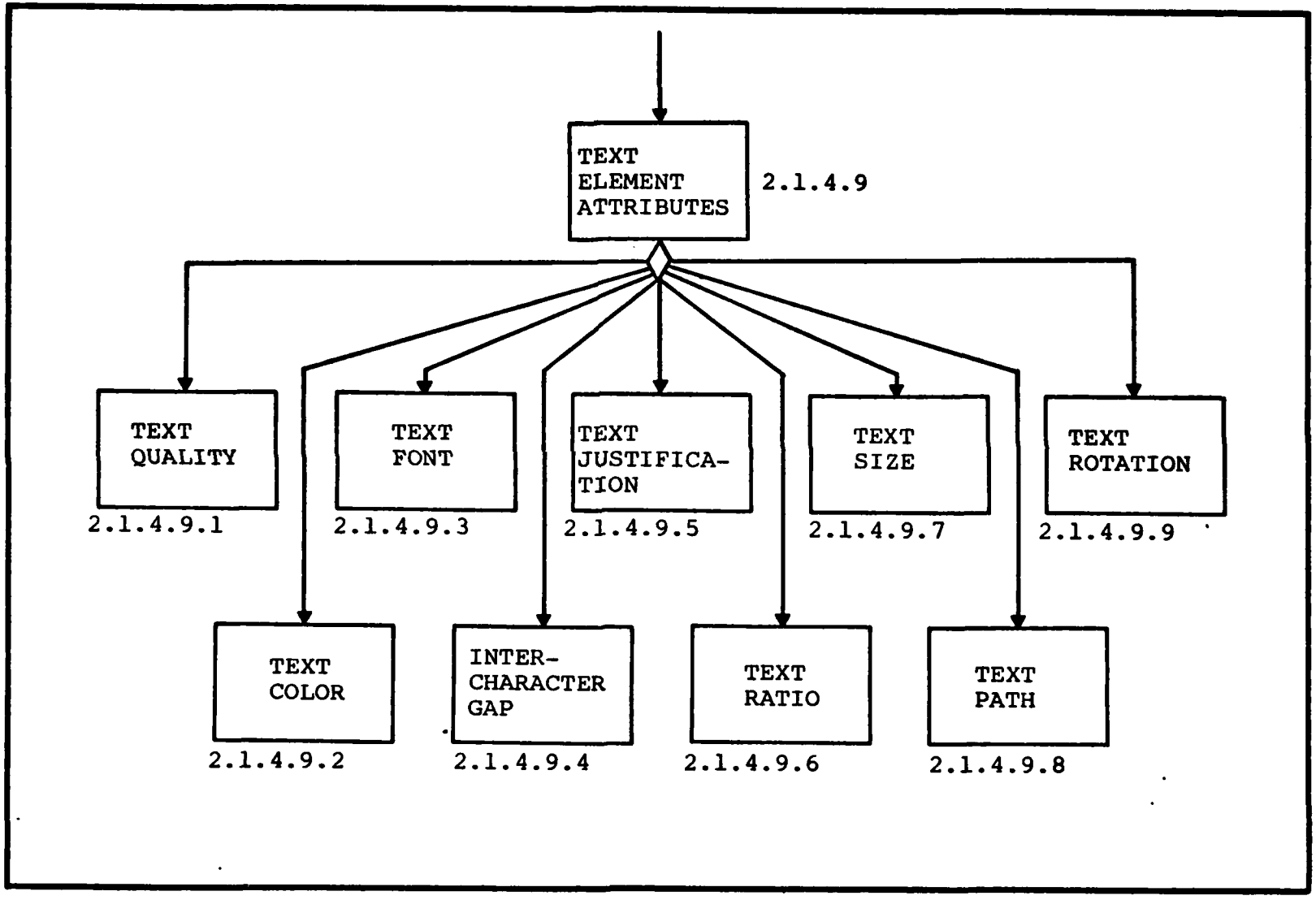


Figure 24. Attributes of Text Graph Elements

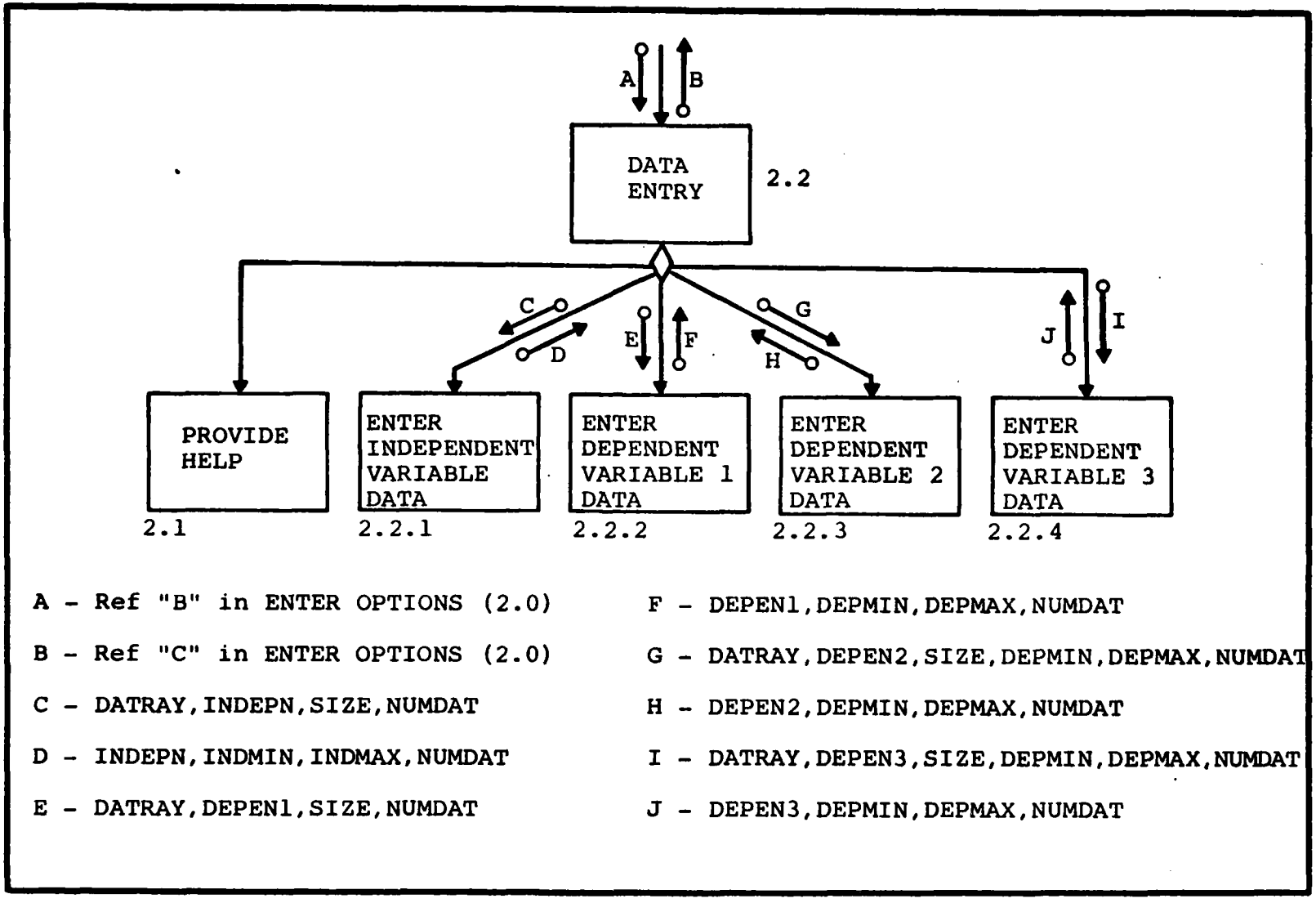


Figure 25. Entering Data Into EZDRAW Data Arrays

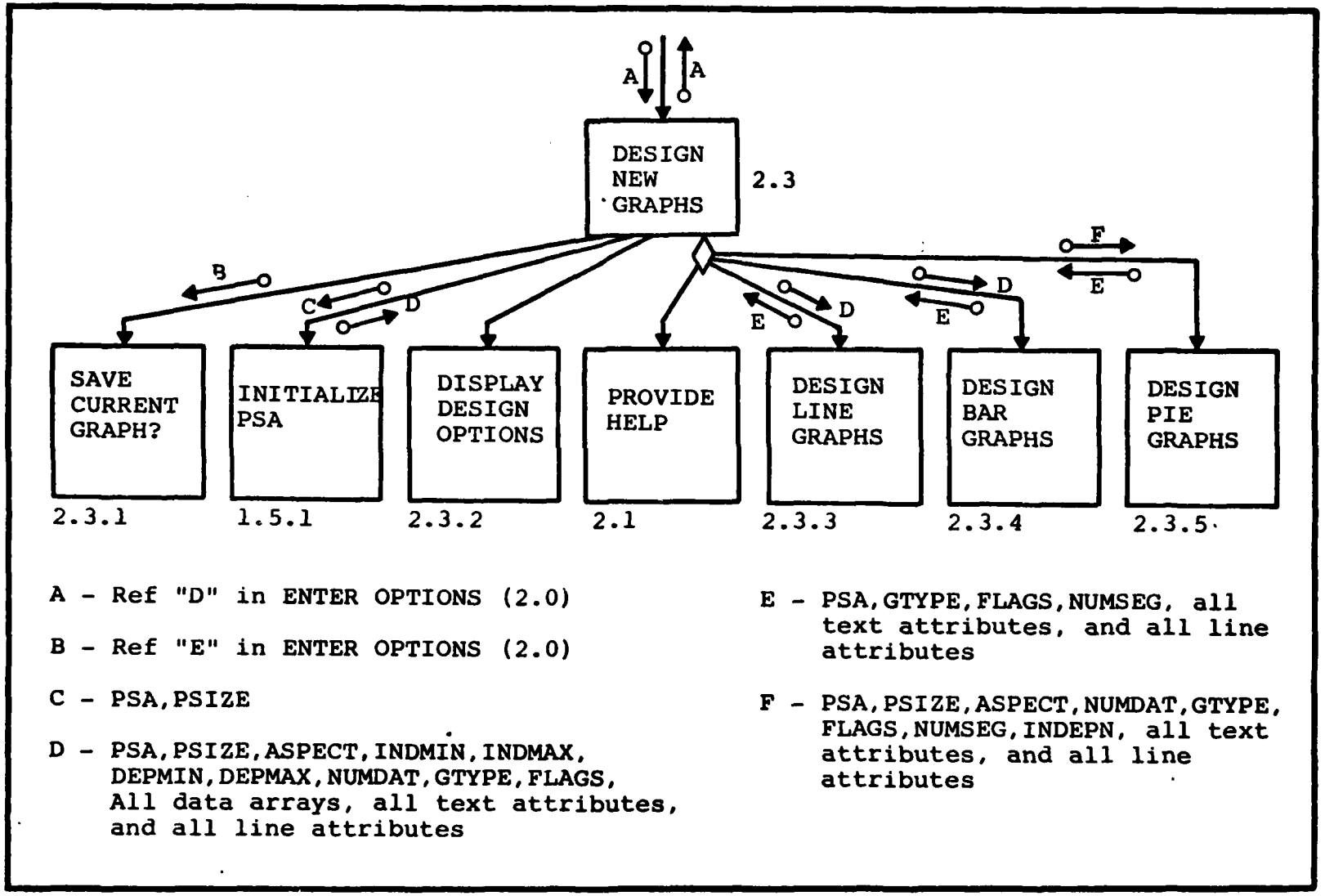


Figure 26. Designing Line, Bar, or Pie Graphs

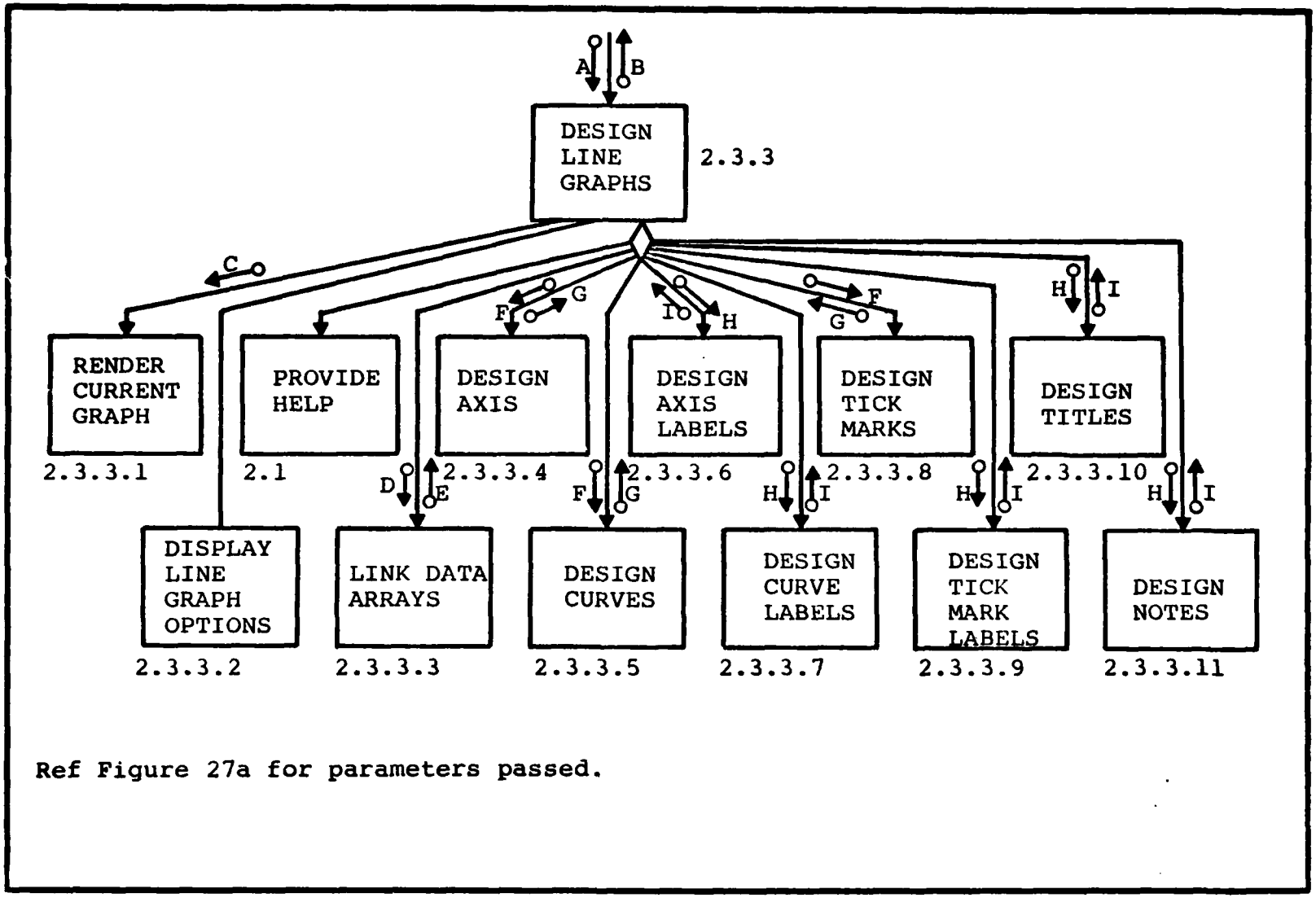


Figure 27. Selecting Which Line Graph Element will be Specified

A - Ref "D" in DESIGN GRAPHS (2.3)

B - Ref "E" in DESIGN GRAPHS (2.3)

C - PSA,PSIZE

D - PSA,PSIZE,INDEPN,DEPEN1,DEPEN2,DEPEN3,INDMIN,
INDMAX,DEPMIN,DEPMAX,NUMDAT,FLAGS

E - PSA,FLAGS

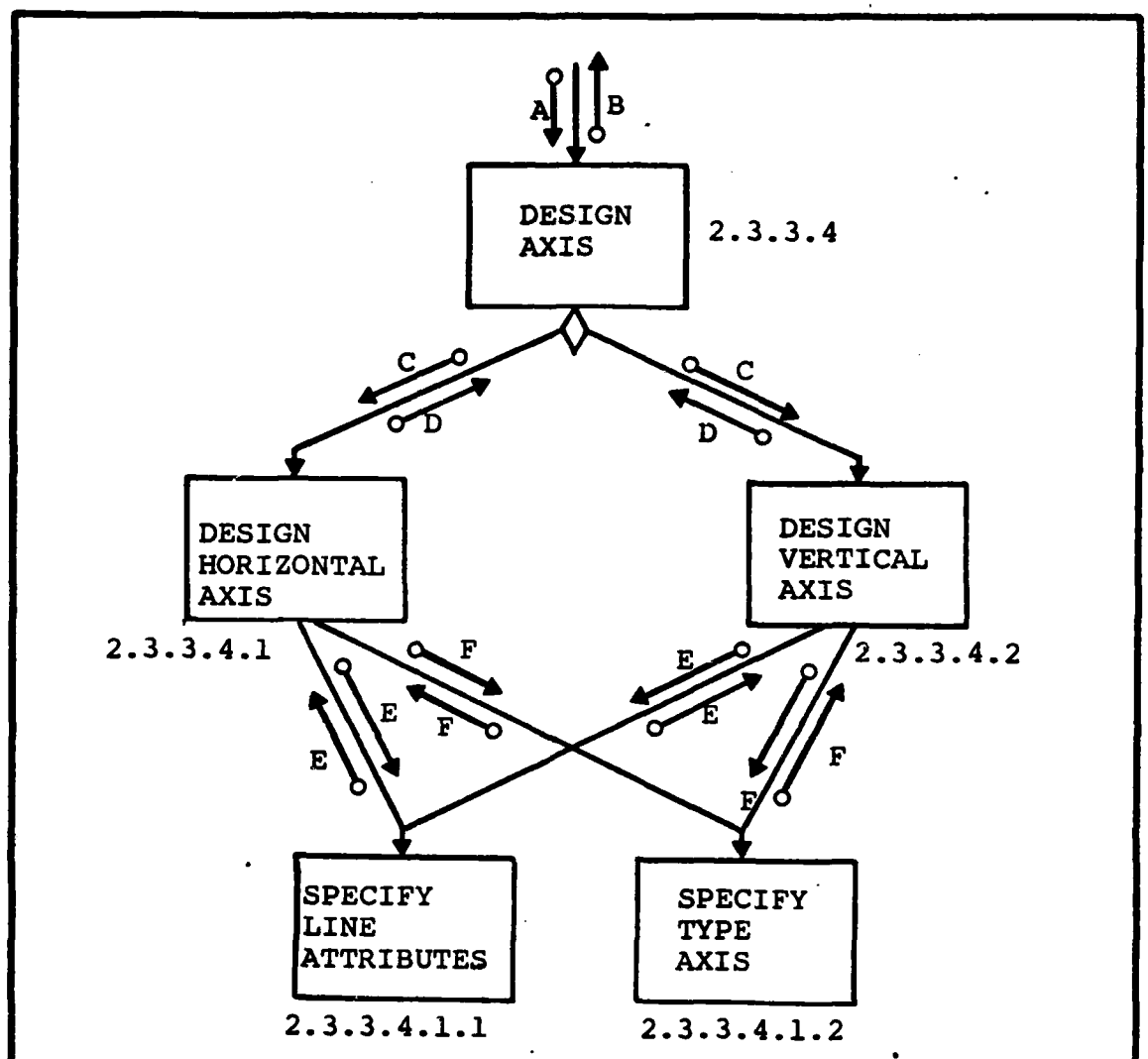
F - PSA,PSIZE,FLAGS, all line attributes

G - PSA,FLAGS, all line attributes

H - PSA,PSIZE,FLAGS, all text attributes

I - PSA,FLAGS, all text attributes

Figure 27a. Line Graph Parameters



A - Ref "F" in DESIGN LINE GRAPHS (2.3.3

B - Ref "G" in DESIGN LINE GRAPHS (2.3.3)

C - PSA,PSIZE, all line attributes

D - PSA, all line attributes

E - All line attributes

F - AXTYPE

Figure 28. Designing Axes

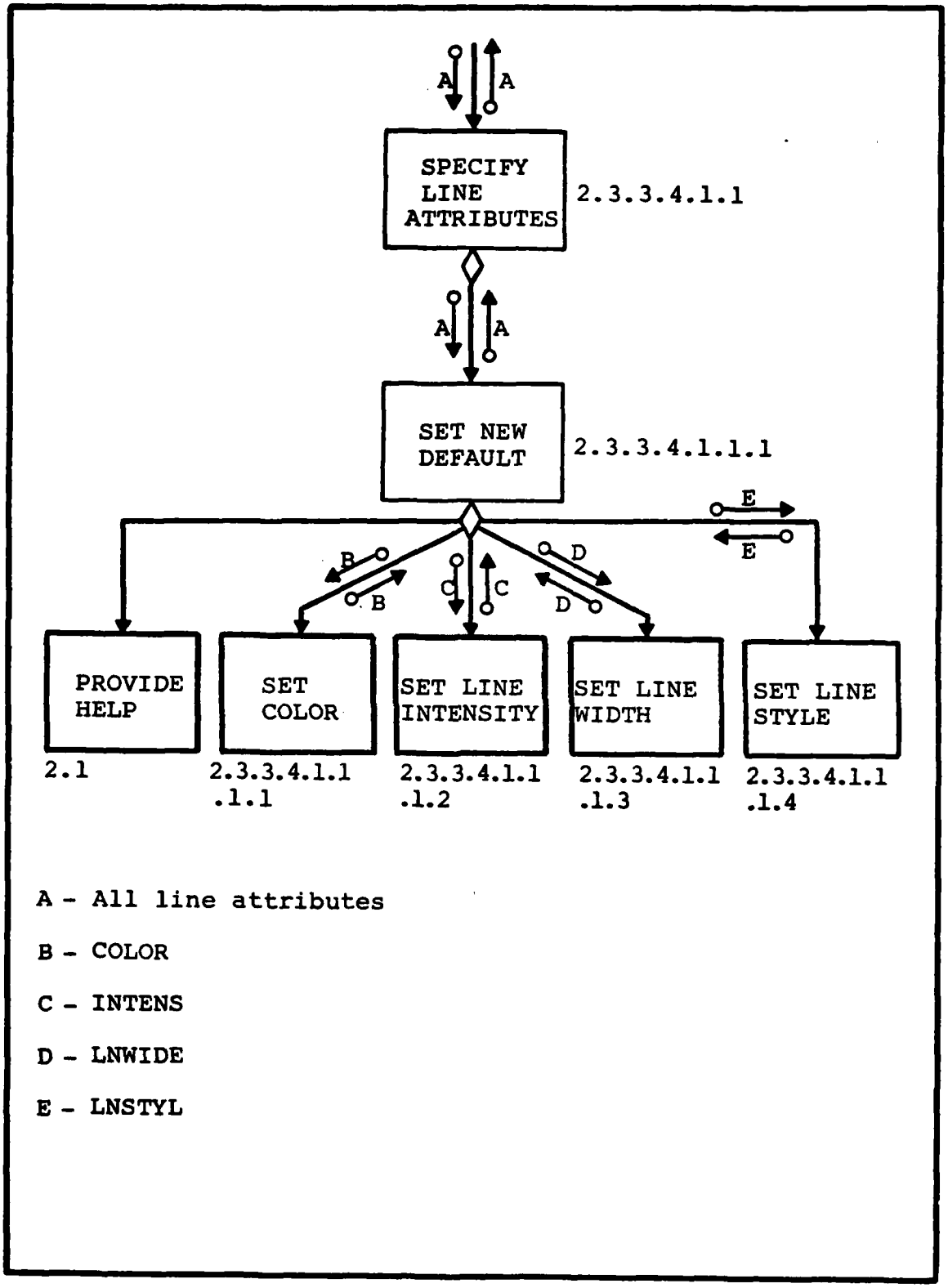


Figure 29. Specifying the Attributes for Linear Graph Elements

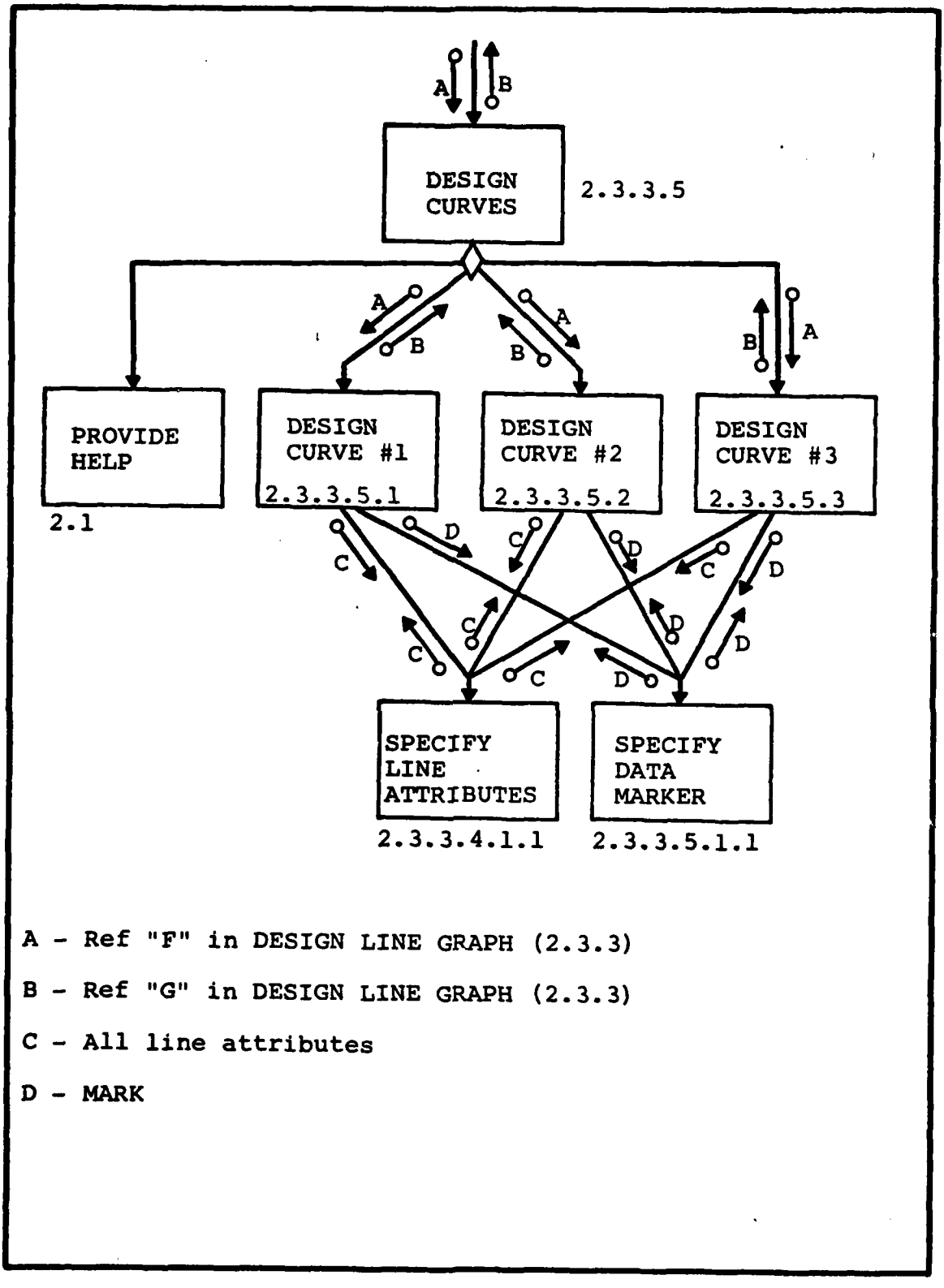


Figure 30. Designing the Curves of a Line Graph

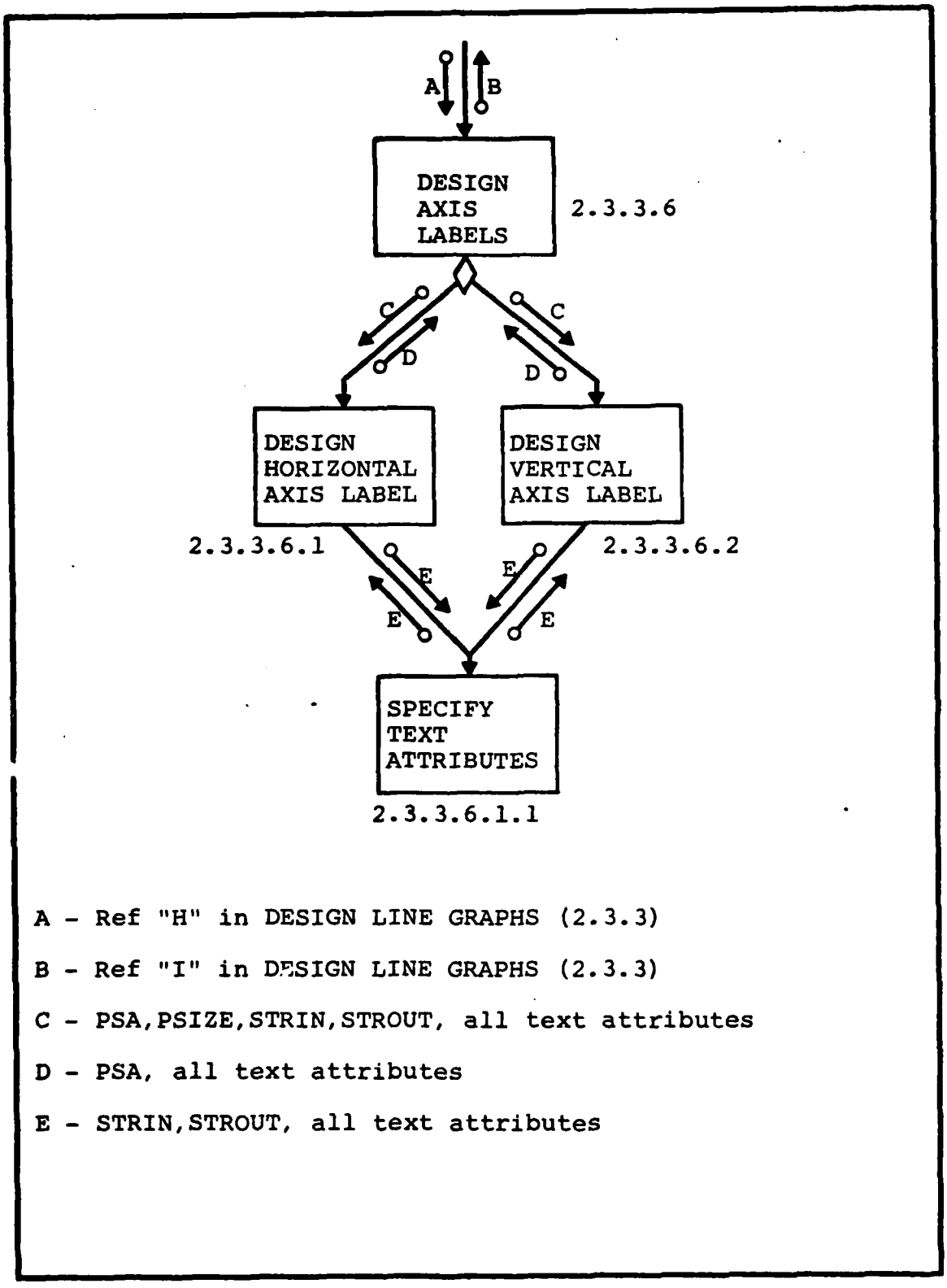


Figure 31. Designing Axes Labels

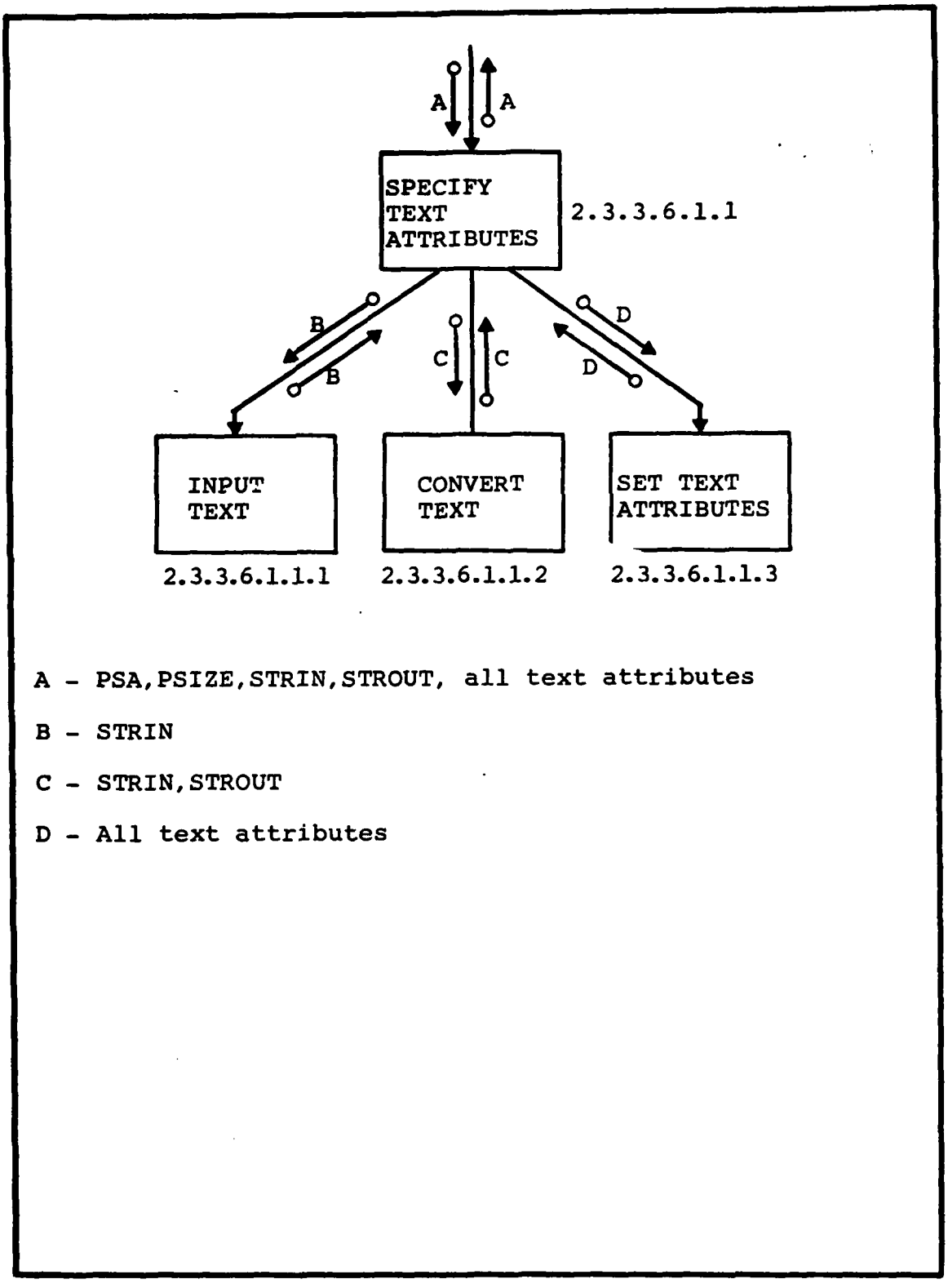


Figure 32. Manipulat..ng Text Strings

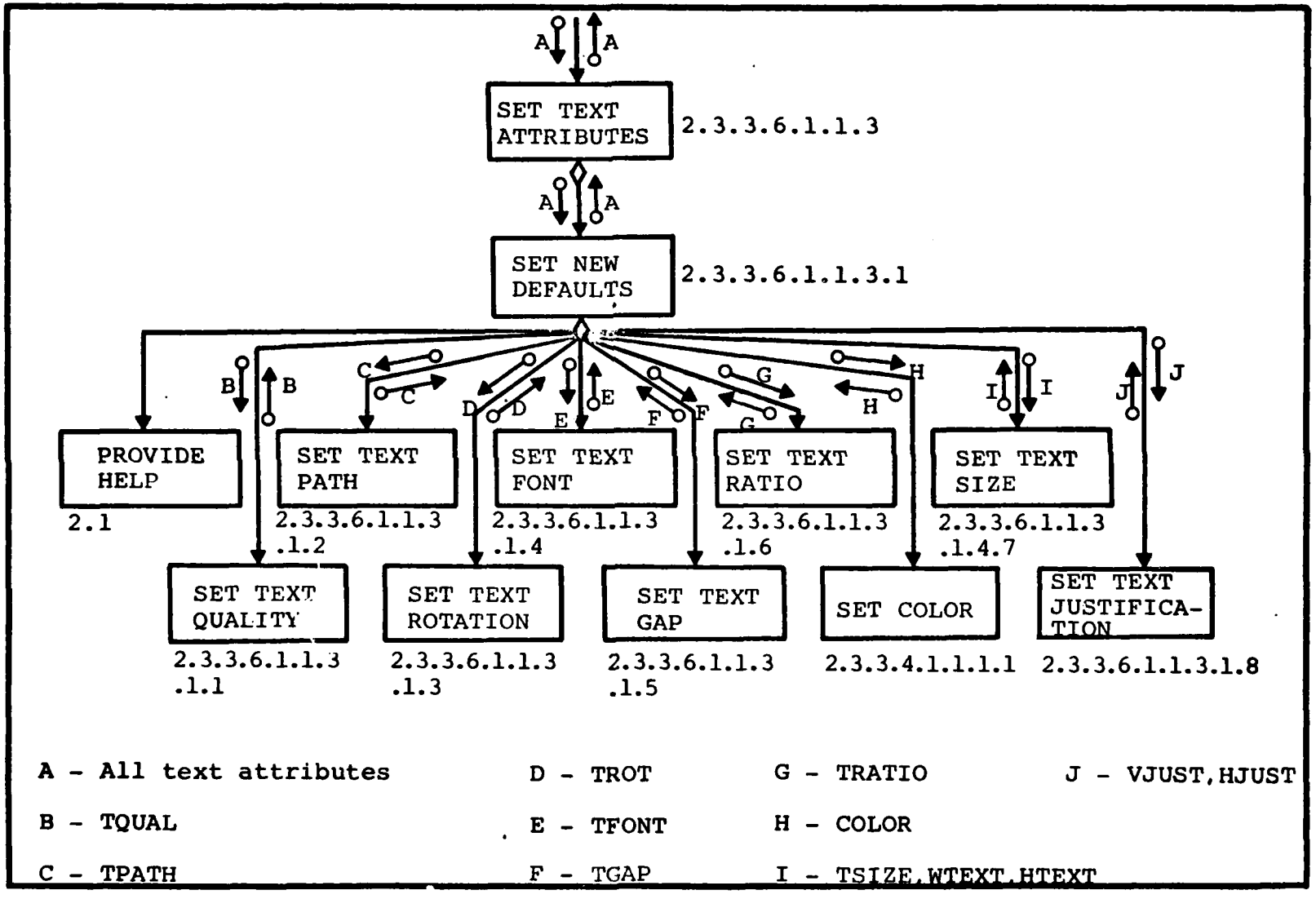
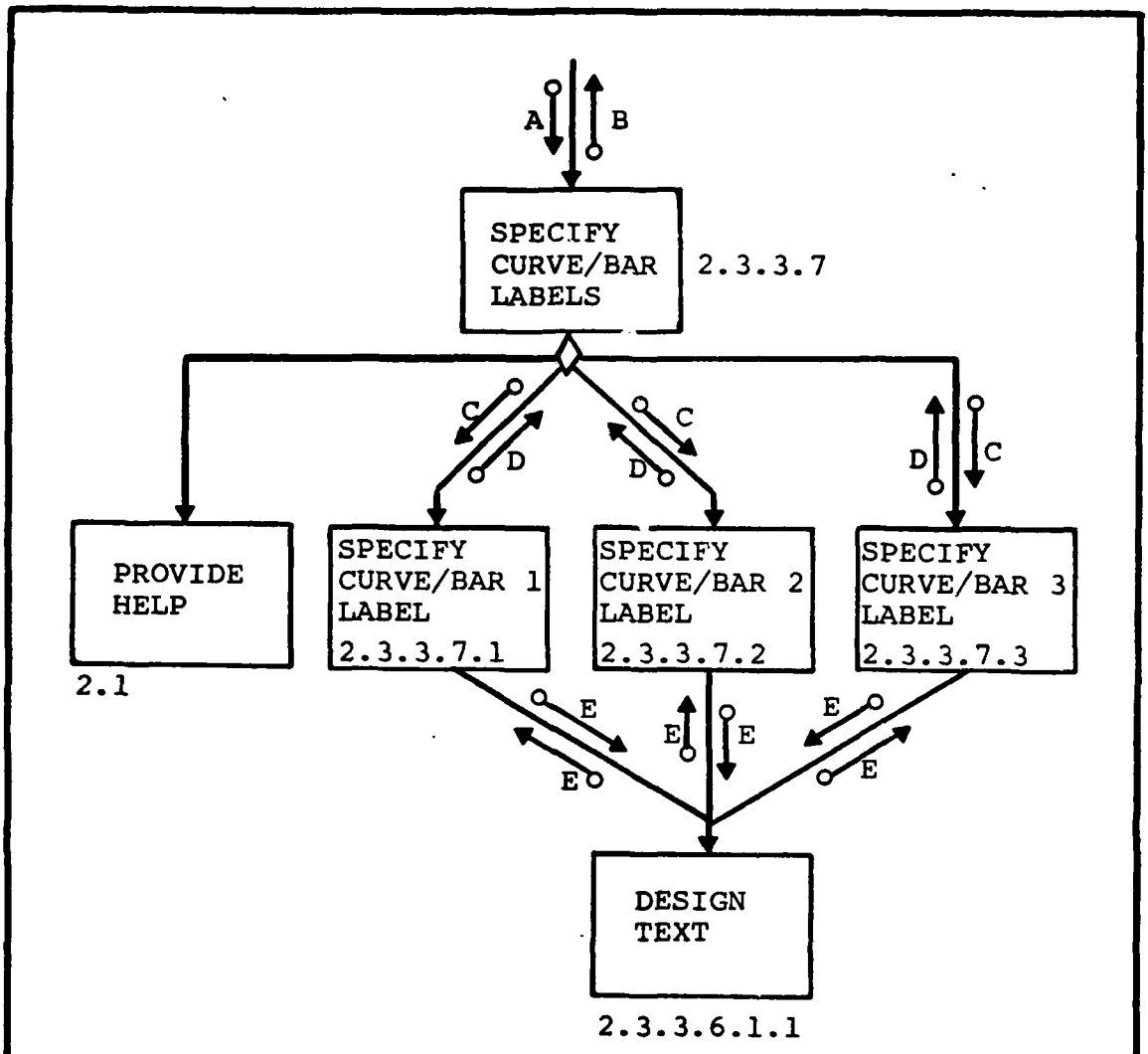


Figure 33. Setting the Text String Attributes



A - Ref "H" in DESIGN LINE GRAPH (2.3.3)

B - Ref "I" in DESIGN LINE GRAPH (2.3.3)

C - PSA,PSIZE,STRIN,STROUT, all text attributes

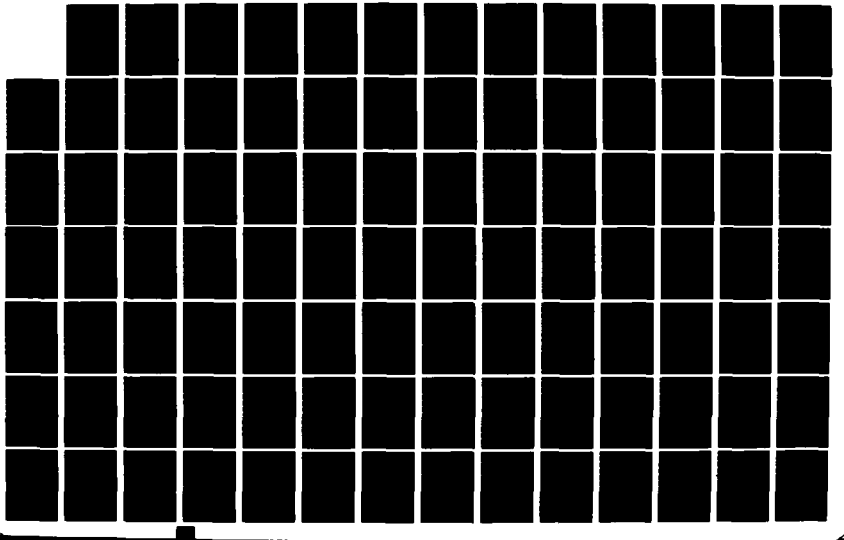
D - PSA, all text attributes

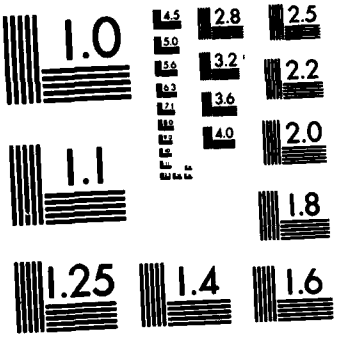
E - STRIN,STROUT, all text attributes

Figure 34. Designing Curve/Bar Labels

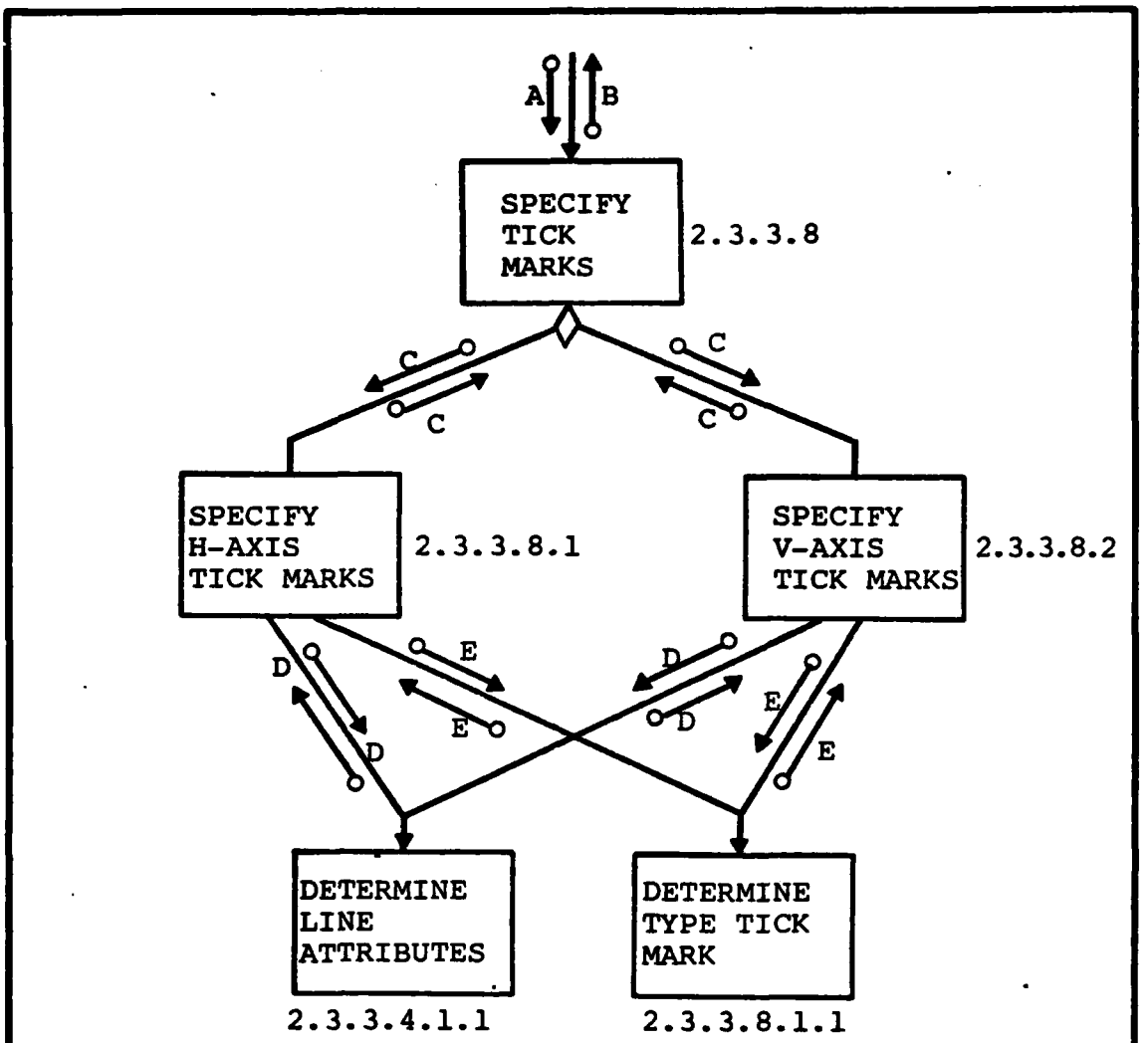
AD-A124 901 EZDRAW AN INTERACTIVE COMPUTER GRAPHICS PROGRAM TO DESIGN BAR LINE OR PIE GRAPHS(U) AIR FORCE INST OF TECH WRIGHT-PATTERSON AFB OH SCHOOL OF ENGI.. R VEA DEC 82 2/3

UNCLASSIFIED AFIT/GE/MA/82D-1 F/G 9/2 NL





MICROCOPY RESOLUTION TEST CHART
NATIONAL BUREAU OF STANDARDS-1963-A



A - Ref "F" in DESIGN LINE GRAPH (2.3.3)

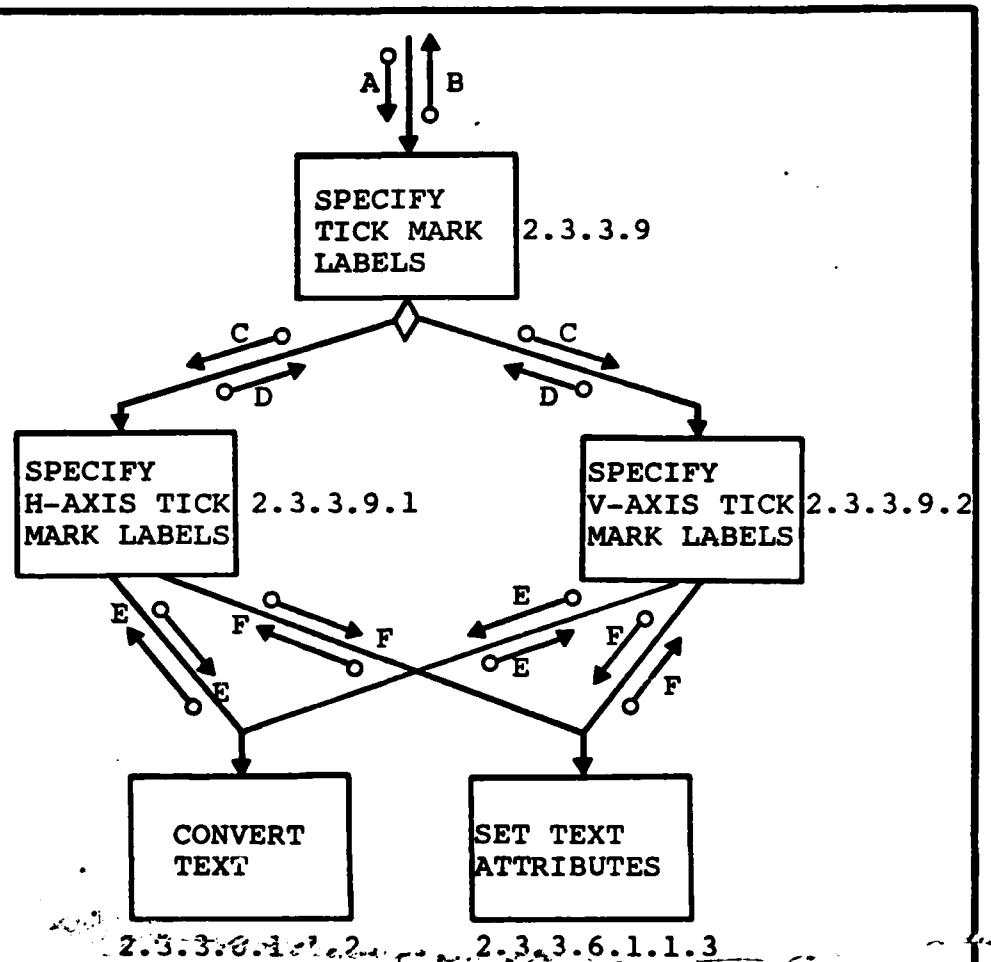
B - Ref "G" in DESIGN LINE GRAPH (2.3.3)

C - PSA,PSIZE, all line attributes

D - All line attributes

E - TICTYP

Figure 35. Designing Tick Marks



A - Ref "H" in DESIGN LINE GRAPH (2.3.3)

B - Ref "I" in DESIGN LINE GRAPH (2.3.3)

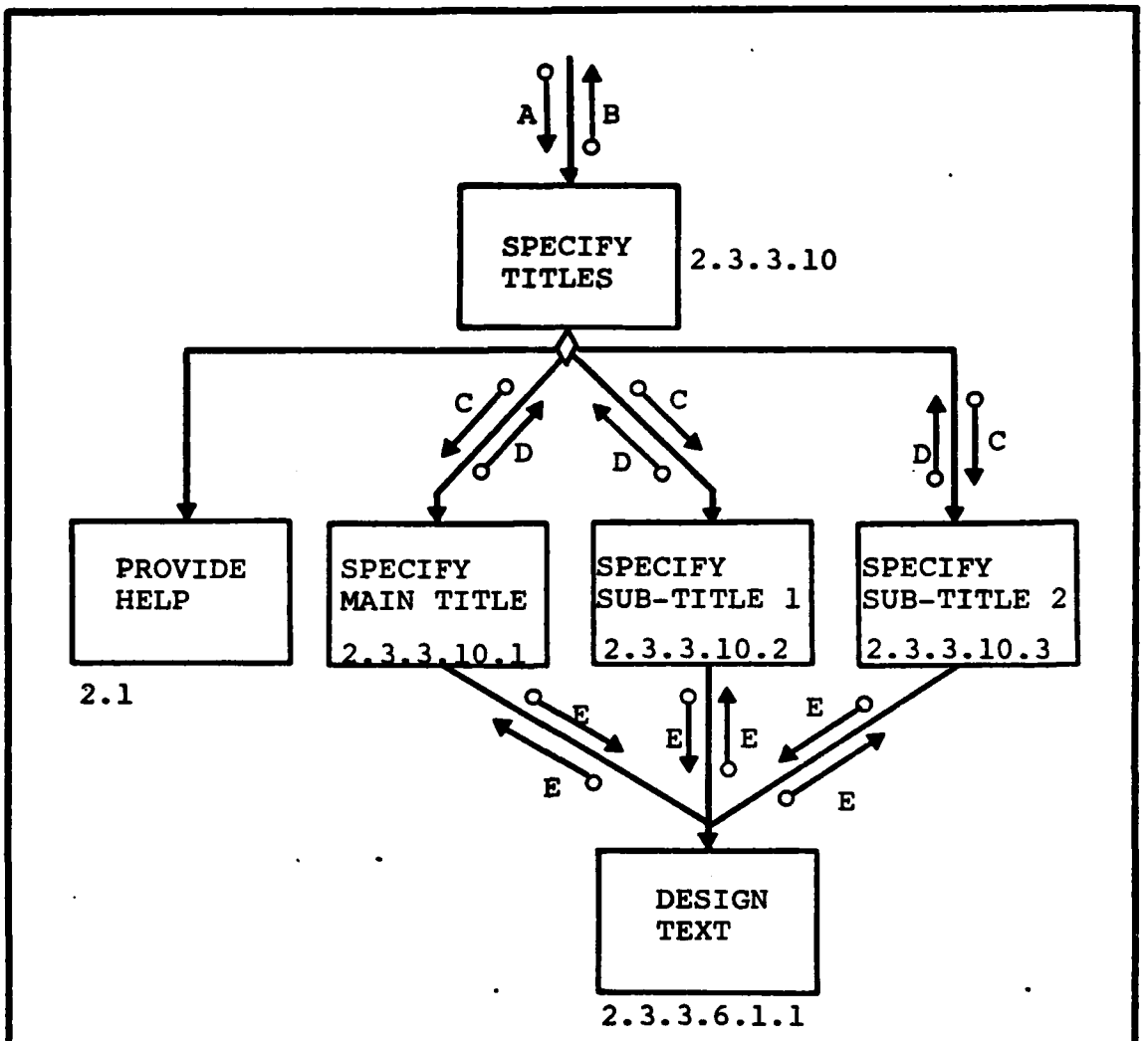
C - PSA,PSIZE,STRIN,STROUT, all text attributes

D - PSA, all text attributes

E - STRIN,STROUT

F - All text attributes

Figure 36. Designing Tick Mark Labels



A - Ref "H" in DESIGN LINE GRAPHS (2.3.3)

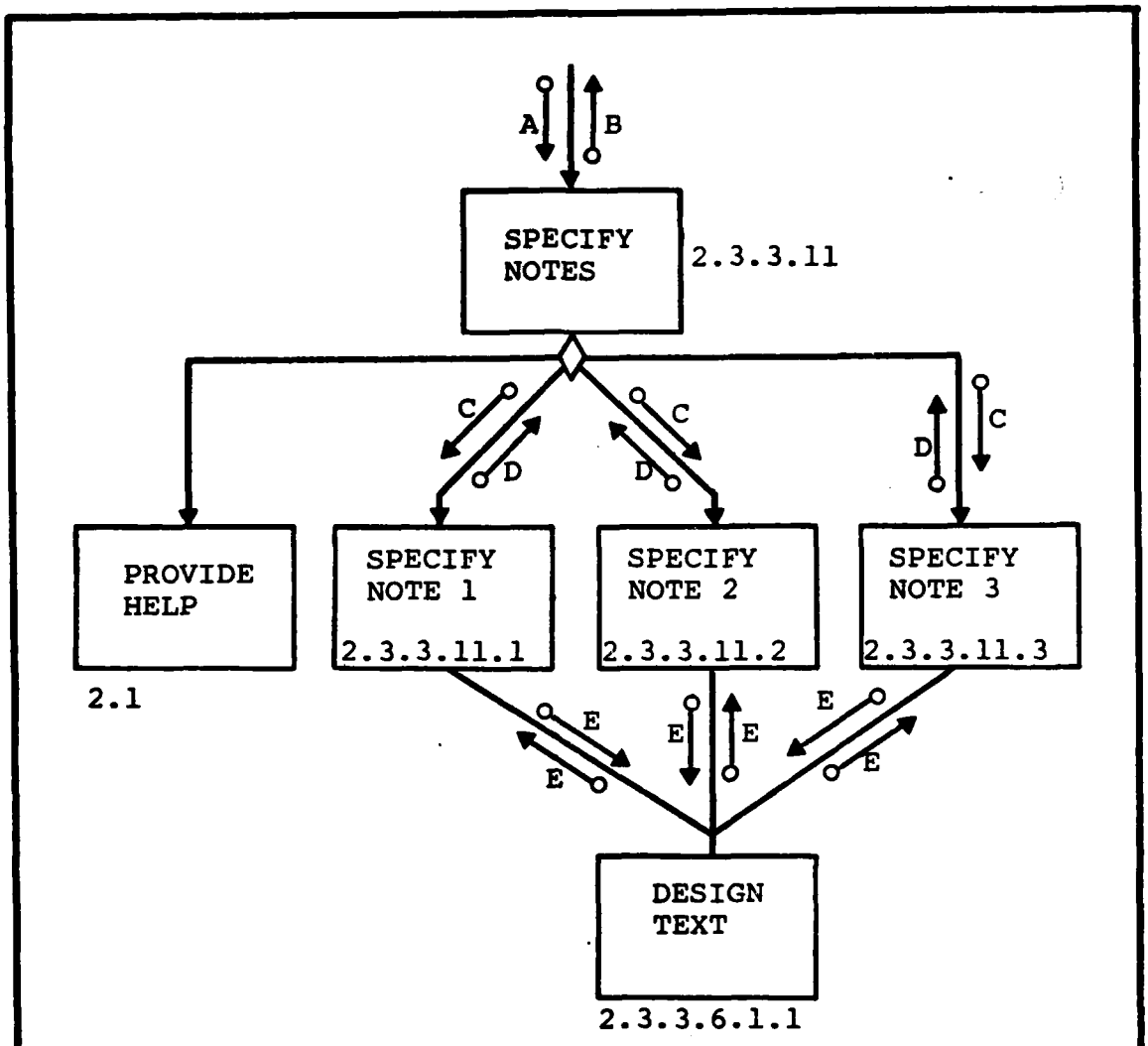
B - Ref "I" in DESIGN LINE GRAPHS (2.3.3)

C - PSA,PSIZE,STRIN,STROUT, all text attributes

D - PSA, all text attributes

E - STRIN,STROUT, all text attributes

Figure 37. Designing Graph Titles



A - Ref "H" in DESIGN LINE GRAPH (2.3.3)

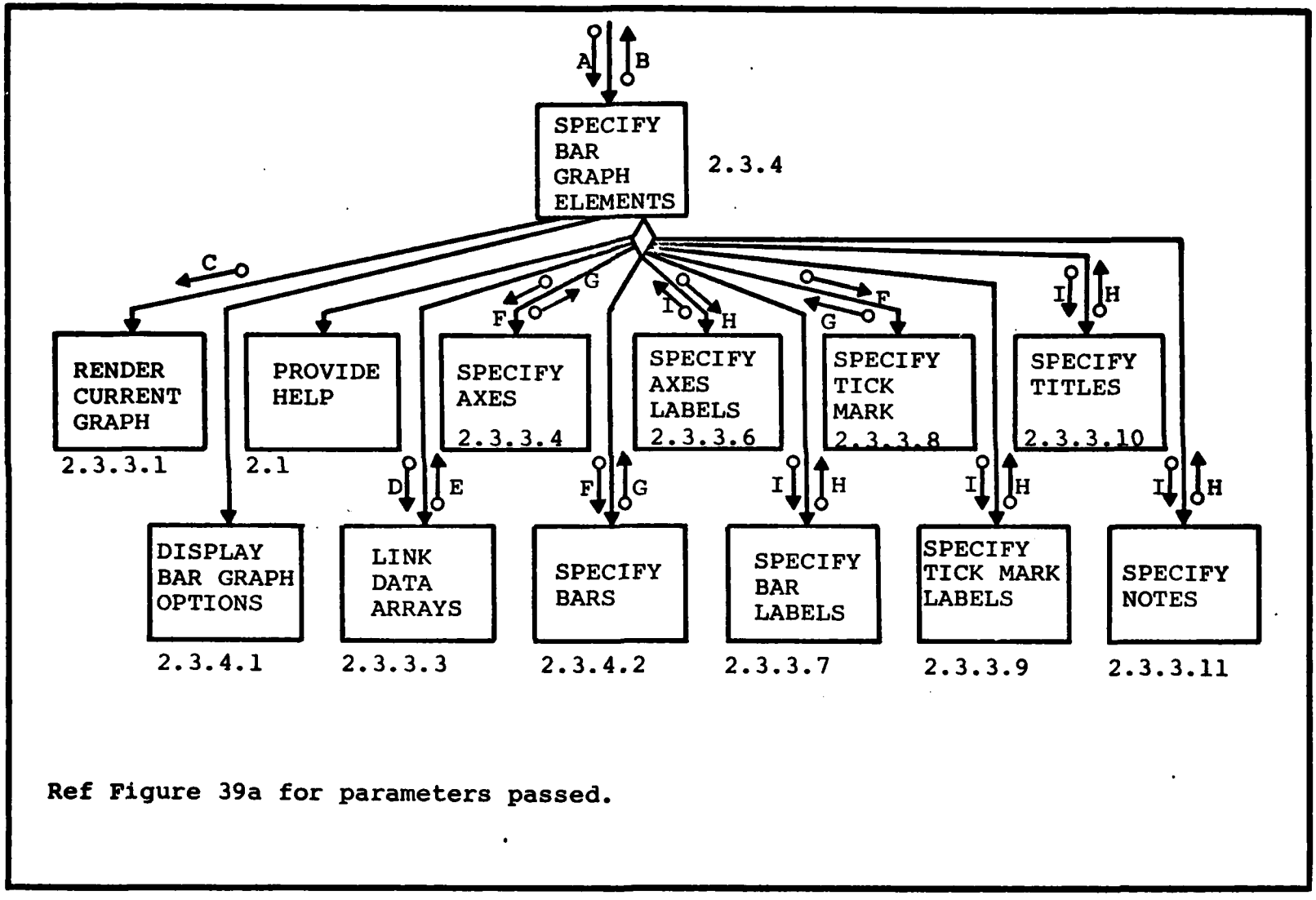
B - Ref "I" in DESIGN LINE GRAPH (2.3.3)

C - PSA,PSIZE,STRIN,STROUT, all text attributes

D - PSA, all text attributes

E - STRIN,STROUT, all text attributes

Figure 38. Designing Additional Graph Notes



Ref Figure 39a for parameters passed.

Figure 39. Selecting Which Bar Graph Element Will be Specified

A - Ref "D" in DESIGN GRAPH (2.3)

B - Ref "E" in DESIGN GRAPH (2.3)

C - PSA,PSIZE

D - PSA,PSIZE,INDMIN,INDMAX,DEPMIN,DEPMAX,NUMDAT, FLAGS, all data arrays

E - PSA,FLAGS

F - PSA,PSIZE,FLAGS, all line attributes

G - PSA,FLAGS, all line attributes

H - PSA,PSIZE,FLAGS, all text attributes

I - PSA,FLAGS, all text attributes

Figure 39a. Bar Graph Parameters

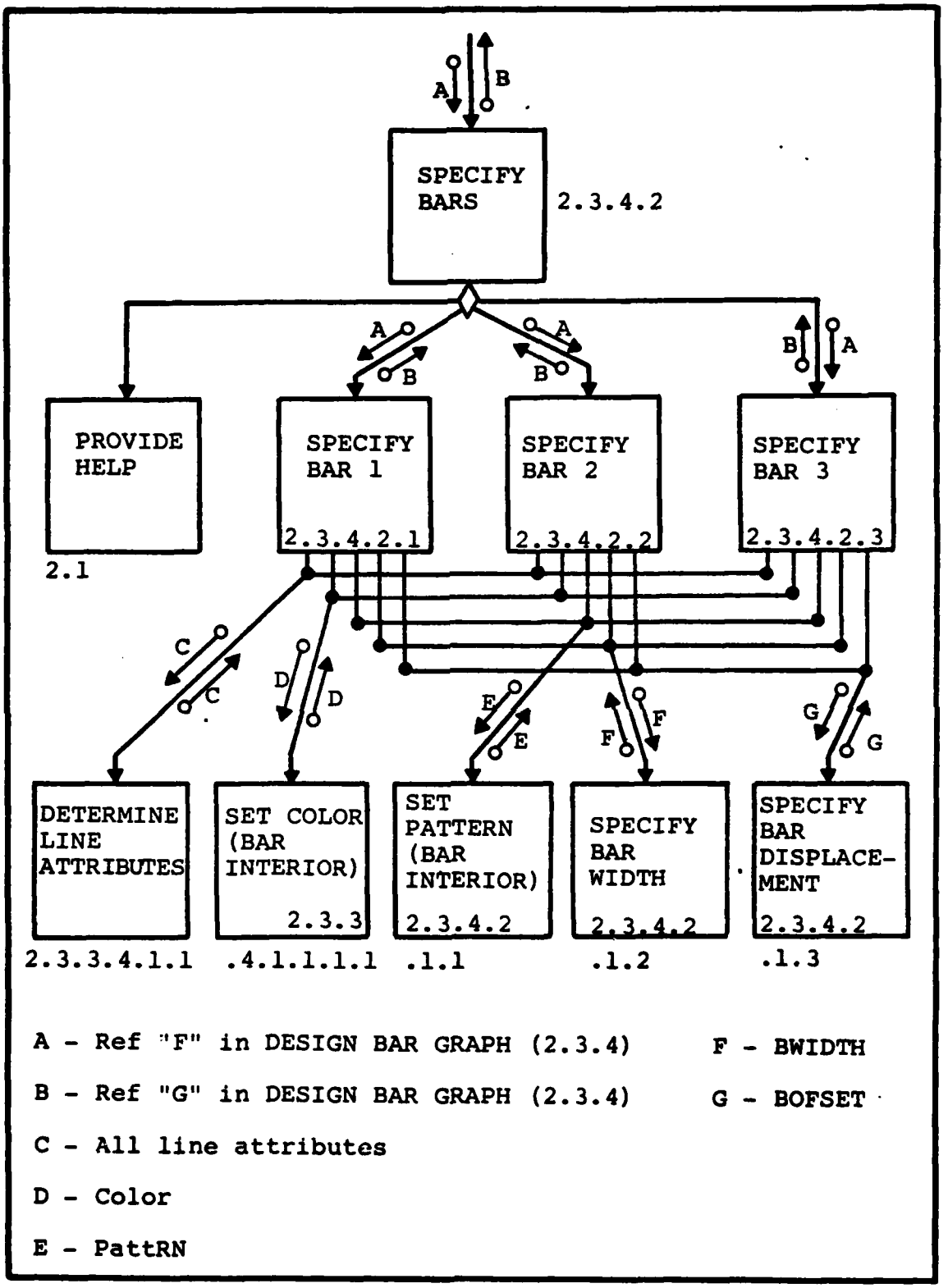


Figure 40. Designing the Bars of a Bar Graph

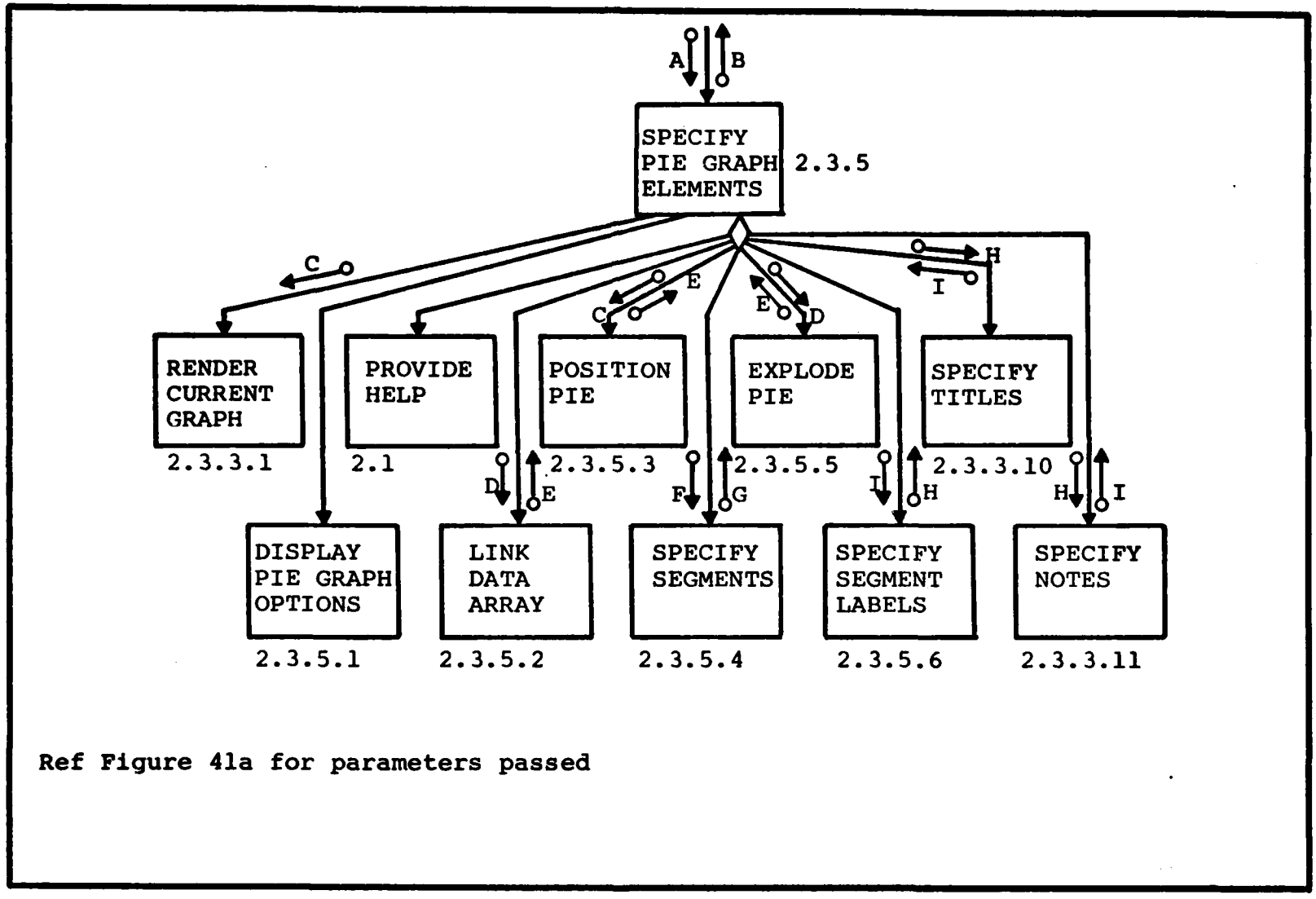


Figure 41. Selecting Which Pie Graph Element Will be Specified

A - Ref "F" in DESIGN GRAPHS (2.3)

B - Ref "E" in DESIGN GRAPHS (2.3)

C - PSA,PSIZE

D - PSA,PSIZE,INDEPN,NUMDAT,NUMSEG,FLAGS

E - PSA,FLAGS,NUMSEG

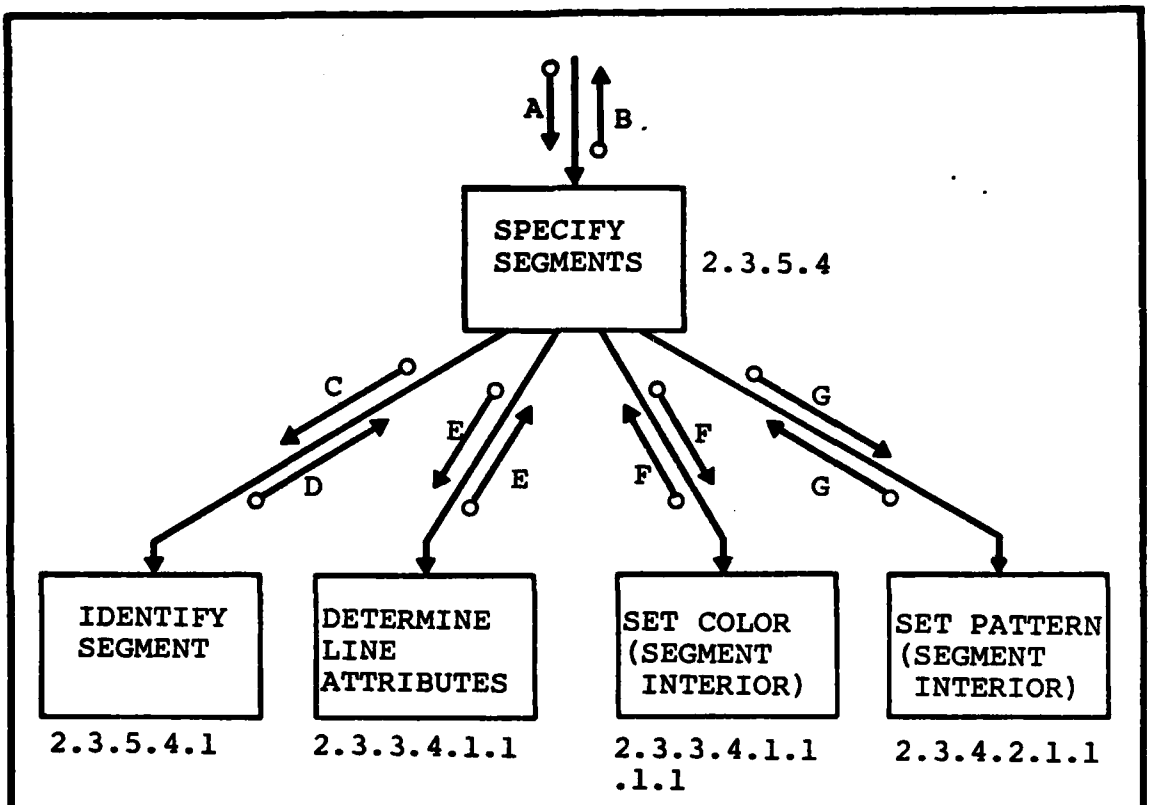
F - PSA,PSIZE,NUMDAT,NUMSEG,FLAGS, all line attributes

G - PSA,NUMSEG,FLAGS, all line attributes

H - PSA,PSIZE,NUMSEG,FLAGS, all text attributes

I - PSA,FLAGS, all text attributes

Figure 41a. Pie Graph Parameters



A - Ref "F" in DESIGN PIE GRAPH (2.3.5)

B - Ref "G" in DESIGN PIE GRAPH (2.3.5)

C - SEGNUM,NUMSEG

D - SEGNUM

E - All line attributes

F - COLOR

G - PATTRN

Figure 42. Designing Pie Graph Segments

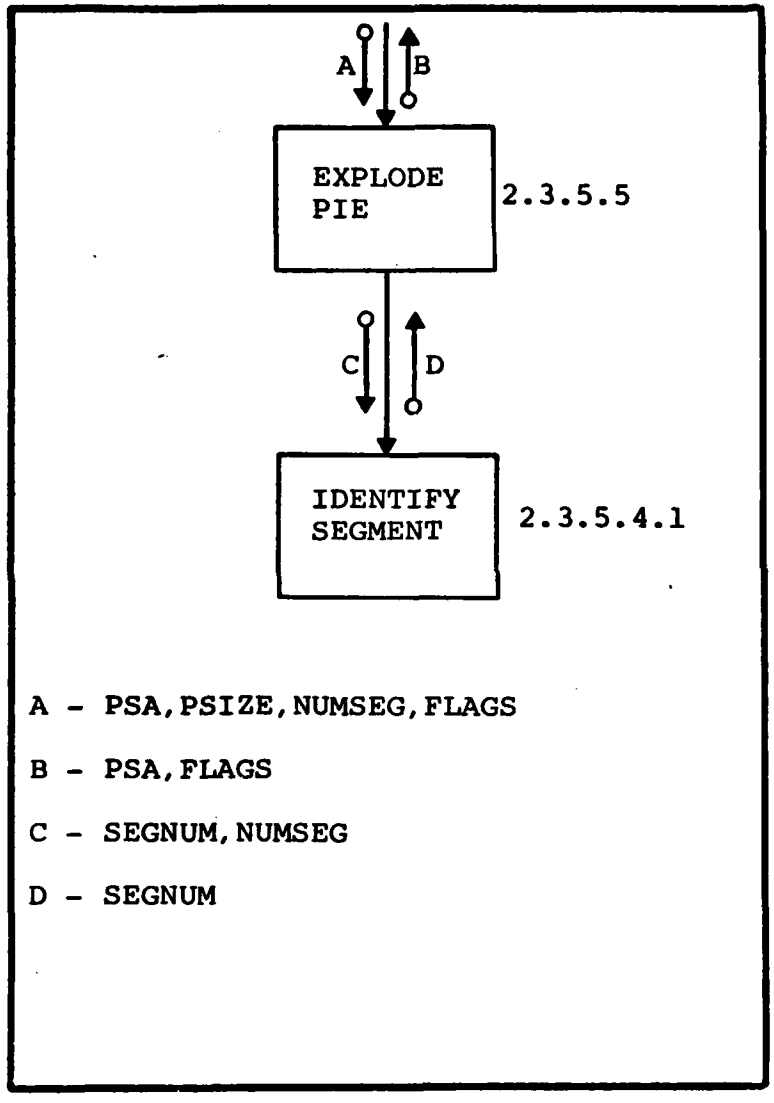


Figure 43. Exploding Pie Segments

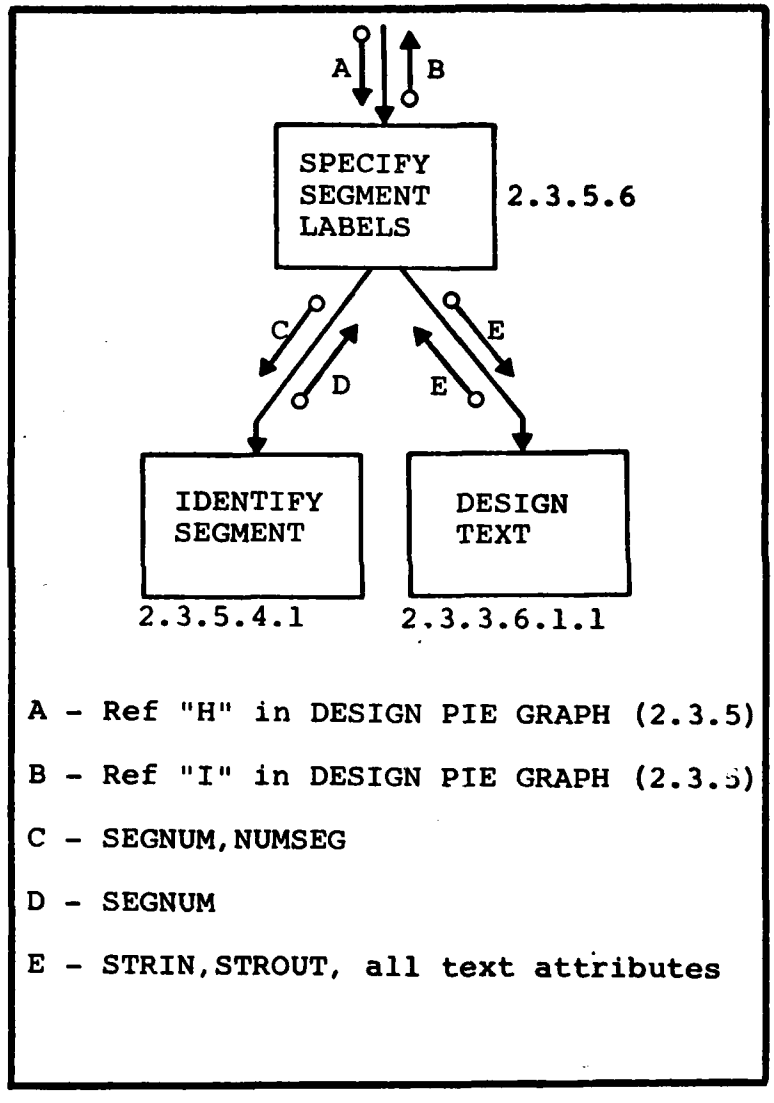
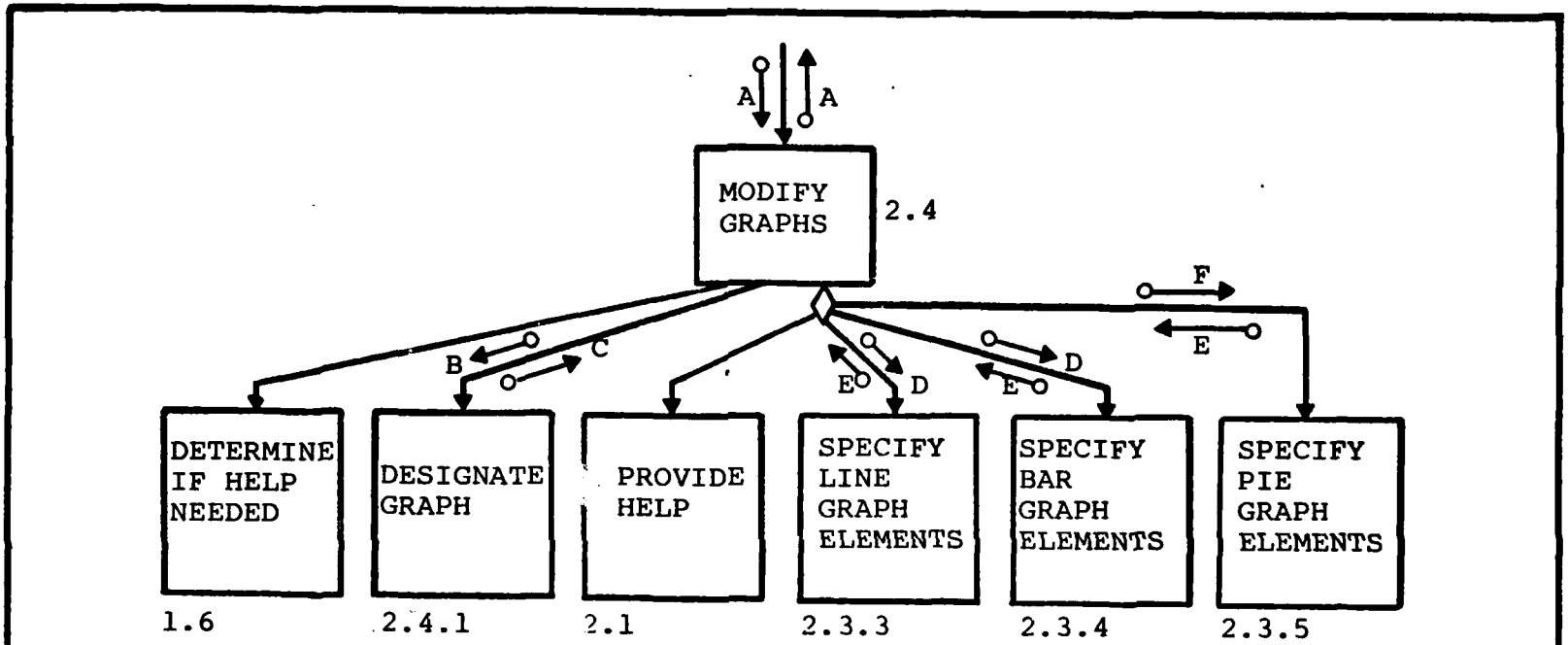


Figure 44. Labelling Pie Segments



A - Ref "D" in ENTER OPTIONS (2.0)

B - PSA,PSIZE,RCDLEN,ASPECT,FLAGS,GTYPE

C - PSA,FLAGS,GTYPE

D - PSA,PSIZE,ASPECT,INDMIN,INDMAX,DEPMIN,DEPMAX,FLAGS, all data arrays, all line attributes, all text attributes

E - PSA,FLAGS, all text attributes

F - PSA,PSIZE,INDEPN,ASPECT,FLAGS,NUMSEG, all text attributes

Figure 45. Modifying Graphs

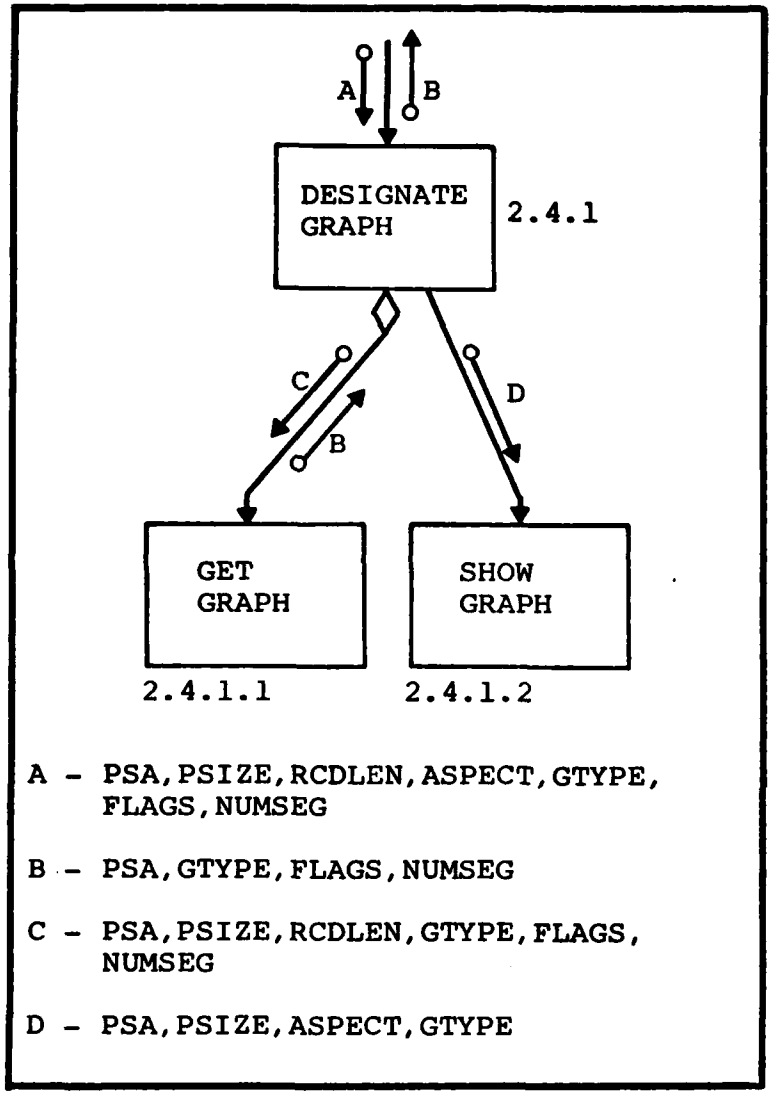


Figure 46. Retrieving Graphs

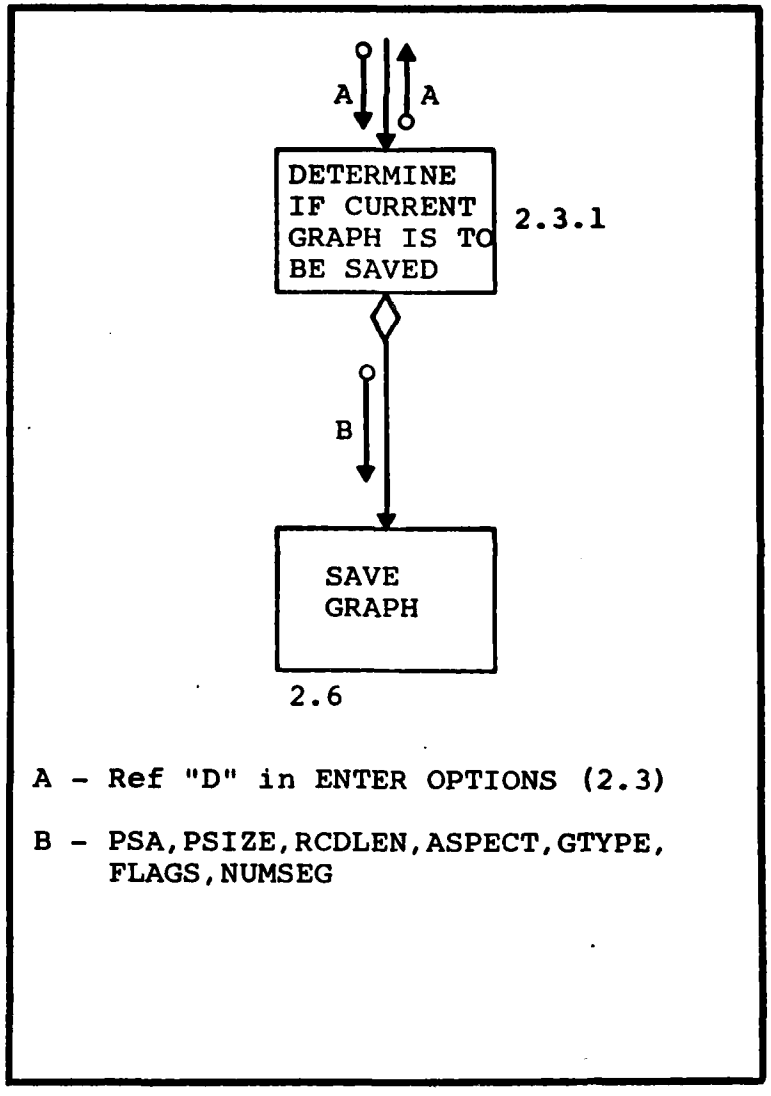


Figure 47. Determining if a Graph is to be Saved

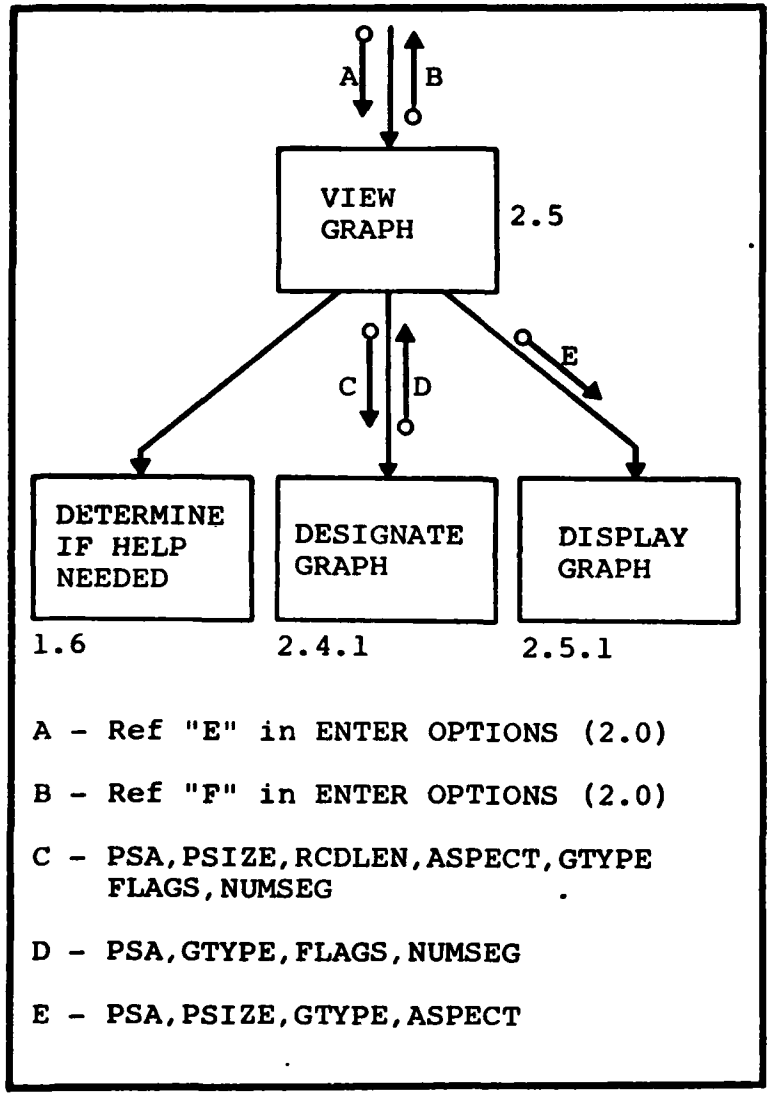


Figure 48. Viewing Graphs

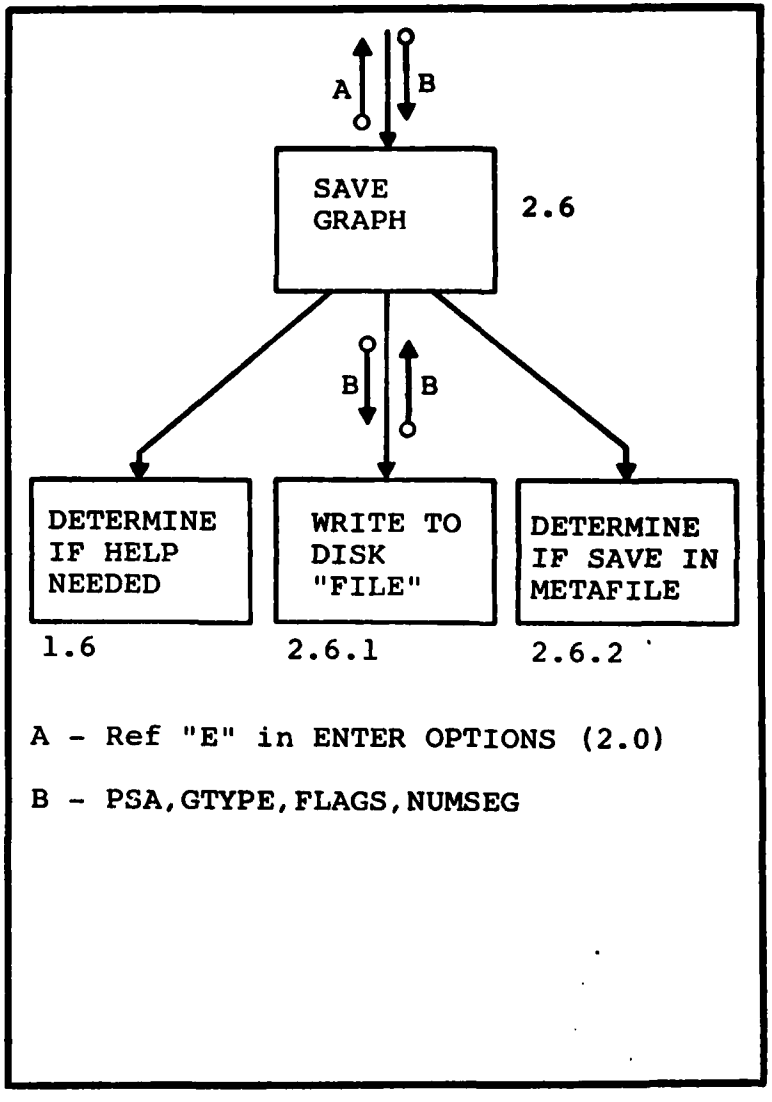
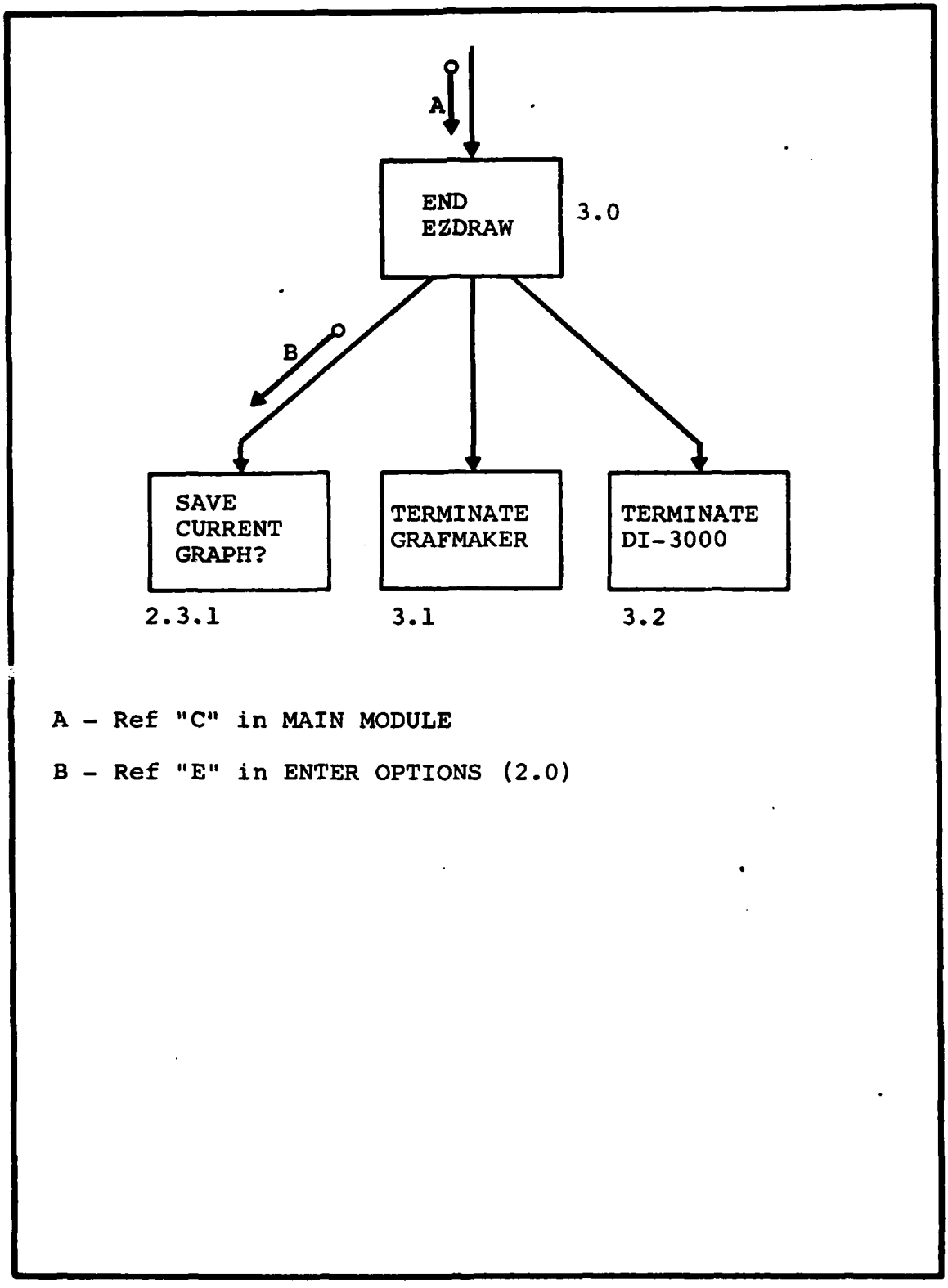


Figure 49. Saving Graphs



A - Ref "C" in MAIN MODULE

B - Ref "E" in ENTER OPTIONS (2.0)

Figure 50. Terminating EZDRAW

Appendix C

Data Dictionary

TABLE II - EZDRAW Variables

NAME	TYPE	REMARKS
ANSWER	CHARACTER*1	A specific response to an EZDRAW prompt
ASPECT	REAL (PARAMETER)	ASPECT [ratio] Display device view space ratio (vertical:horizontal)
AXTYPE	INTEGER	AXis TYPE Type of axis (linear/log/ln) for horizontal or vertical axes of line or bar graphs Ref G:A-6
BOFSET	REAL	Bar OFfSET Displacement of a bar group from an axis tick mark [left/below (-) or right/above (+)] in data units Local:BAR1, BAR2, BAR3, BARDIS Ref G:A-10
BWIDTH	REAL	Bar WIDTH Width of a bar group in data units Local: BAR1, BAR2, BAR3, BARWID Ref G:A-10
CHOICE	INTEGER	User response to an EZDRAW menu
COLOR	INTEGER	Index describing the color of a graph element Ref G:A-42
CTINUE	CHARACTER*1	ConTINUE Response to message prompt to continue processing

CYCLE	INTEGER	Index describing which tick mark(s) along a line or bar graph axis will be labelled Local: HTCLBL, VTCLBL Ref G:A-75
DATRAY	REAL ARRAY DIMENSION(SIZE)	DATa arRAY General purpose array wherein data is input interactively or from an external data file prior to storing in graph data arrays
DEPEN1	REAL ARRAY DIMENSION(SIZE)	DEPENdent [data array] 1 Data array for a dependent data set linked to GRAFMAKER as curve/bar #1 Ref G:A-35
DEPEN2	REAL ARRAY DIMENSION(SIZE)	DEPENdent [data array] 2 Data array for a dependent data set linked to GRAFMAKER as curve/bar #2 Ref G:A-35
DEPEN3	REAL ARRAY DIMENSION(SIZE)	DEPENdent [data array] 3 Data array for a dependent data set linked to GRAFMAKER as curve/bar #3 Ref G:A-35
DEPMAX	REAL	DEPendent [variable] MAXimum [value] Maximum data value of the dependent variable
DEPMIN	REAL	DEPendent [variable] MINimum [value] Minimum data value of the dependent variable
DEVNUM	INTEGER PARAMETER	DEVice NUMber Physical display device for GRAFMAKER output Ref G:A-24

DIR	INTEGER	DIRection Index specifying whether pie graph segments are drawn clock-wise or counter-clock-wise Local: PIPOSN Ref G:A-50
DPNEND	REAL	DePeNdent [axis] END [value] Specifies ending value of the dependent axis in data units Local: DATLNK Ref G:A-48, A-90
DPNSTR	REAL	DePeNdent [axis] STaRt [value] Specifies starting value of the dependent axis in data units Local: DATLNK Ref G:A-48, A-90
FILNAM	CHARACTER*12	FILe NAMe Name of external data/graph file
FLAGS	INTEGER ARRAY DIMENSION(55)	A system of status flags to indicate whether a graph element has been previously specified
FSEG	INTEGER	First [exploded] SEGment Specifies which pie segment will be the first segment of a contiguous, exploded group of pie segments Local: PEXPLO Ref G:A-62
GTYPE	INTEGER	Graph TYPE An index denoting whether a graph is a line (1), bar (2), or pie (3) graph
HJUST	INTEGER	Horizontal [text] JUSTification Index describing horizontal justification point for the text string Ref G:A-83

HTEXT	REAL	Height (of) TEXT Specifies vertical "buffer" around text character Ref G:A-82, A-89
IDAT	INTEGER	Integer DATa Integer data input interactively or from data file that is converted to real data then stored in the real data arrays
INDEND	REAL	INDependent [axis] END [value] Specifies ending value of the independent axis in data units Local: DATLNK Ref G:A-48, A-90
INDEPN	REAL ARRAY DIMENSION(SIZE)	INDEPeNdent [data array] Data array for independent variable (line/bar or pie graphs)
INDMAX	REAL	INDependent [variable] MAXimum [value] Maximum data value of the independent variable
INDMIN	REAL	INDependent [variable] MINimum [value] Minimum data value of the independent variable
INDSTR	REAL	INDependent [axis] STaRt [value] Specifies the starting value of the independent axis in data units Local: DATLNK Ref G:A-48, A-90
INTENS	INTEGER	[line] INTENSity Index specifying a relative intensity (brightness) of a graphical line element Ref G:A-94
LENGTH	INTEGER	Specifies the length of an input text string w/o the trailing blanks

LINDEX	INTEGER	Line INDEX [number] Index associated with the attributes of the "line" which delineates a pie segment Local: PISGMT Ref G:A-61
LNSTYL	INTEGER	LiNe STYLe Index specifying the appearance of a graphical line element Ref G:A-94
LNWIDE	INTEGER	LiNe WIDE (width) Index specifying the relative width (thickness) of a line graph element Ref G:A-94
LSEG	INTEGER	Last [exploded] SEGment Specifies which pie segment will be the last segment of a contiguous, exploded group of pie segments Local: PEXPLO Ref G:A-62
LSTRT	INTEGER	Label STaRT Specifies which axis tick mark will receive the first tick mark label Local: HTCLBL, VTCLBL Ref G:A-75
MARK	INTEGER	[data point] MARKer Index specifying the type of data point marker to be used on curves for line graphs Ref G:A-40
MAXVAL	REAL	MAXimum VALue The maximum data value of a data set input interactively or from a data file into DATRAY
MINVAL	REAL	MINimum VALue The minimum data value of a data set input interactively or from a data file into DATRAY

NUMDAT	INTEGER	NUMber [of] DATa Specifies the number of data values to be entered interactively or from a data file into DATRAY
NUMLBL	INTEGER	NUMber [tick mark] LaBeLs Specifies how many different tick mark labels have been entered as a tick mark label group for a line/bar graph axis Local: HTCLBL, VTCLBL Ref G:A-73
NUMSEG	INTEGER	NUMber [of pie] SEGments Specifies how many data points were passed to GRAFMAKER as pie segment values
NUSTRG	CHARACTER*82	New [text] STRinG The input text string with the EZDRAW delimiters concatenated before and after the last non-blank character. Used primarily in STRMKR
PANGLE	REAL	Pie ANGLE Angle measured clock-wise or counter-clock-wise from the horizontal to where the first pie graph segment will be drawn Local: PIPOSN Ref G:A-56
PATTRN	INTEGER	[polygon interior fill] PATTeRN Index describing a device dependent pattern to fill the interior of pie segments and bar groups Ref G:A-42
PSA	INTEGER ARRAY DIMENSION(PSIZE)	Picture Storage Area Integer array that stores the "graphical" data; must be dimensioned to at least 1500 decimal words Ref G:A-18

PSIZE	INTEGER PARAMETER	Picture [storage area] SIZE Specifies the PSA dimension Ref G:A-18
RADIUS	REAL	Defines radius of a pie graph in chart coordinates Local: PIPOSN Ref G:A-56
RCDLEN	INTEGER PARAMETER	[file] ReCorD LENgth Size of a record within a direct access file
RCDNUM	INTEGER	[file] ReCorD NUMber Specific record within an existing direct access file
RDAT	REAL	Real DATa Real data input interactively or from a data file into DATRAY then stored into a real data array
SEGNUM	INTEGER	[pie] SEGment NUMber Identifies a particular segment within a pie graph Local: PISGMT, SEGLBL, PEXPLO, PICSEG
SIZE	INTEGER PARAMETER	Determines the dimension of the data arrays and the number of data values that can be input interactively or from a data file
STRIN	CHARACTER*80	[text] STRing IN Character array into which all graph text strings are input; This FORTRAN 77 variable must be converted to an equivalent integer variable in order to be accepted by GRAFMAKER Ref G:5-6

STROUT	INTEGER ARRAY DIMENSION(21)	[text] STRing OUT Integer array wherein the converted character variable STRIN is stored through an internal read and then is passed to GRAFMAKER as the character string/"ADNOT" Ref G:A-2
TANGLE	REAL	Text ANGLE An angle in which a specified text string is rotated about its justification point either clock-wise or counter-clock-wise Ref G:A-87
TCINCR	REAL	TiCk [mark] INCRement Interval between tick marks along an axis in data units Local: HTIC, VTIC Ref G:A-78
TCSTRT	REAL	TiCk [mark] STaRT [point] Data value along an axis where the first tick mark will be located Local: HTIC, VTIC Ref G:A-78
TFONT	INTEGER	Text FONT Index describing the print/character style to be used for a graph text string Ref G:A-95, D:34-35.2
TGAP	REAL	Text GAP Index describing the inter-character gap to be used for a graph text string Ref G:A-95, D:36
TICEND	REAL	TICk [mark] END [point] Data value along an axis where the last tick mark will be located Local: HTIC, VTIC Ref G:A-78

TICTYP	REAL	TICk [mark] TYPe Index describing whether the graph tick mark is to be a major/minor tick mark or major/minor grid line Ref G:A-77
TPATH	INTEGER	Text PATH Index describing whether the text string will be written "right, left, up, or down" Ref G:A-84
TQUAL	INTEGER	Text QUALity Index describing the precision with which text strings are generated and the degree to which they conform to other text attributes Ref G:A-85
TRATIO	REAL	Text RATIO A ratio of character width to character height; Set to 0.75 Ref G:A-95
TSIZE	REAL	Text SIZE The height of a character in chart coordinates; Does not include the character "buffer" surrounding each character as part of a text string box Ref G:A-82
VJUST	INTEGER	Vertical JUSTification Index specifying the point within the text string box where the justification point is located vertically Ref G:A-83
WTEXT	REAL	Width [of] TEXT [character] A percentage of the character height in chart coordinates which specifies the horizontal "buffer" between each character of a text string Ref G:A-82, A-89

XFCTR	REAL	eXplode FaCToR A percentage of the pie graph radius (in chart coordinates) that exploded pie segments are detached from the rest of the pie; Set to 0.20 in EZDRAW Local: PEXPLO Ref G:A-62
XO	REAL	Horizontal X-[chart] coordinate that specifies the position of a text string relative to the justification point of the string Local: POSTXT Ref G:A-58
XPIE	REAL	X [coordinate] PIE Specifies the location of the center of the pie graph in chart coordinates in the horizontal direction Local: PIPOSN Ref G:A-56
XV	REAL ARRAY DIMENSION(4)	X Virtual [coordinate] Used by DI-3000 in specifying world coordinates and then by EZDRAW to "draw" a border Ref D:A-64
YO	REAL	Vertical Y-[chart] coordinate that specifies the position of a text string relative to the justification point of the string Local: POSTXT Ref G:A-58
YPIE	REAL	Y [coordinate] PIE Specifies the location of the center of the pie graph in chart coordinates in the vertical direction Local: PIPOSN Ref G:A-56

YV	REAL ARRAY DIMENSION(4)	Y Virtual [coordinate] Used by DI-3000 in specifying world coordinates and then by EZDRAW to "draw" a border Ref D:A-64
ZV	REAL ARRAY DIMENSION(4)	Z Virtual [coordinate] Used by DI-3000 in specifying world coordinates and then by EZDRAW to "draw" a border Ref D:A-64

Appendix D

EZDRAW Subroutine Reference Maps

TABLE III - EZDRAW Subroutines

Name:	AREABM (AREA Bar [graph] Menu)
Function:	Displays the Bar Graph Menu options in a designated viewport of the display screen.
Calls:	JCLOSE, JCRLF, JJUST, JMARGN, JMOVE, JOPEN, JSIZE, J1TEXT
Called By:	BMENU
Name:	AREALM (AREA Line [graph] Menu)
Function:	Displays the Line Graph Menu options in a designated viewport of the display screen.
Calls:	JCLOSE, JCRLF, JJUST, JMARGN, JMOVE, JOPEN, JSIZE, J1TEXT
Called By:	LMENU
Name:	AREAPM (AREA Pie [graph] Menu)
Function:	Displays the Pie Graph Menu options in a designated viewport of the display screen.
Calls:	JCLOSE, JCRLF, JJUST, JMARGN, JMOVE, JOPEN, JSIZE, J1TEXT
Called By:	PMENU
Name:	BAR
Function:	Allows the user to specify the attributes of a particular bar group (i.e. bar width, displacement from axis tick marks, interior color/fill pattern).
Calls:	BAR1, BAR2, BAR3, EMSGM, EZHELP, JCHCLR
Called By:	BARGRF
Name:	BARDAT (BAR DATa)
Function:	Provides information about graph data represented as bars on a bar graph.
Calls:	JCHCLR, EZGO
Called By:	DATRA

Name:	BARDIS (BAR DISplacement)
Function:	Determines the displacement from an axis tick mark for a particular bar group of a bar graph. A bar group may be displaced left or right of a tick mark but displacement is measured in "data units".
Calls:	JCHCLR
Called By:	BAR1, BAR2, BAR3

Name:	BARGRF (BAR GRaph)
Function:	Drives the Bar Graph Menu and accesses all the routines that allow the user to design the elements of a bar graph.
Calls:	BAR, BARLBL, BMENU, DATLNK, EMSGM, EZHELP, GAXES, GAXLBL, JCHCLR, JCHSHW, NOTES, TICLBL, TICMRK, TITLE
Called By:	DESIGN, MODIFY

Name:	BARLBL (BAR LaBeL)
Function:	Allows the user to specify for which of three bar groups, a data label will be specified and designed.
Calls:	DLBL1, DLBL2, DLBL3, EMSGM, EZHELP, JCHCLR
Called By:	BARGRF

Name:	BARS
Function:	Provides information about bar graphs.
Calls:	JCJCLR, EZGO
Called By:	DSGINF

Name:	BARWID (BAR WIDth)
Function:	Allows the user to specify the width of a particular bar group in "data units".
Calls:	JCHCLR
Called By:	BAR1, BAR2, BAR3

Name:	BAR1, BAR2, BAR3
Function:	Accesses the EZDRAW routines that allows the user to specify the attributes for each individual bar group of the bar graph. These attributes include interior color/pattern, width, and displacement.
Calls:	BARDIS, BARWID, EZGO, JBRATR, JCHCLR, JDTATR, JDTFIL, JXLINE, LINATR, PLYFIL, SETCLR
Called By:	BAR

Name:	BIGPRT (BIG [view] PoRT)
Function:	Defines the portion of the view space of the display to which the graph will be mapped when the user wants to "view" the graph and/or make a hard copy of the graph (i.e. the viewport that maximizes the screen for graph display).
Calls:	JVPORT, JWINDO
Called By:	SAVE, SHOW

Name:	BMENU (Bar [graph] MENU)
Function:	Displays the user options, in menu format, when specifying the elements of a bar graph.
Calls:	AREABM, BORDER, DPORT, JWCLIP, MPORT, PPORT, PROMPT
Called By:	BARGRF

Name:	BORDER
Function:	Draws a "border" around all viewports defined by EZDRAW by calling the DI-3000 polygon drawing routine(s). Also delineates the graph in accordance with graphics design procedures.
Calls:	JCLOSE, JIWIND, JMOVE, JOPEN, JPOLY
Called By:	BMENU, GETGRF, LMENU, PMENU, SAVE, SHOW, SPORT

Name:	CAUTNS (CAUTioNS)
Function:	Provides information about situations to avoid when saving graphs.
Calls:	EZGO, JCHCLR
Called By:	SAVINF

Name:	CDMENU (CommanD MENU)
Function:	Displays the user options, in menu format, of EZDRAW options (main modules). This is the highest level menu presented to the user.
Calls:	JCHCLR
Called By:	OPTION

Name:	CHKHLP (CHecK HeLP)
Function:	Determines if the user wants to enter the EZDRAW HELP Mode.
Calls:	EMSGA, EZANSR, EZHELP, JCHCLR
Called By:	MODIFY, SAVE, STRTEZ, VIEW

Name:	CHKSAV (CHecK SAVe)
Function:	Determines if the user wants to save the current graph in the GRAFMAKER Picture Storage Area. This option is presented when the Picture Storage Area will be re-initialized for processing of another graph.
Calls:	EMSGA, EZANSR, JCHCLR, SAVE
Called By:	DESIGN, ENDEZ, MODIFY, VIEW

Name:	CRVDAT (CuRVe DATa)
Function:	Provides information about data represented as curves in a line graph.
Calls:	EZGO, JCHCLR
Called By:	DATRA

Name:	CURLBL (CURve LaBeL)
Function:	Displays message to user and then determines which curve of a line graph will have a label specified and designed.
Calls:	DLBL1, DLBL2, DLBL3, EMSGM, EZHELP, JCHCLR
Called By:	LINGRF

Name:	CURVE
Function:	Determines which curve of a line graph will have its attributes specified (i.e. line style, line width, line intensity, color, and data marker).
Calls:	CURVE1, CURVE2, CURVE3, EMSGM, EZHELP, JCHCLR
Called By:	LINGRF

Name:	CURVE1, CURVE2, CURVE3
Function:	Accesses the EZDRAW routines that allow the user to specify the attributes for a particular curve of a line graph.
Calls:	EZGO, JCHCLR, JDTATR, JXLINE, LINATR, LINMRK
Called By:	CURVE

Name:	DATAIN (DATA INput)
Function:	Allows the user to input line/bar/pie graph data interactively or from an external data file. One independent data set and three dependent data sets can be input to EZDRAW. The number of data points in the data set(s) is determined by the EZDRAW parameter "SIZE".
Calls:	DATPRO, DATPR1, DATPR2, DATPR3, EMSGM, EZGO, EZHELP, JCHCLR
Called By:	OPTION

Name:	DATFIL (DATa FILe)
Function:	Provides information about external data files used by EZDRAW.
Calls:	EZGO, JCHCLR
Called By:	DATINF

Name:	DATINF (DATa INFormation)
Function:	Provides information about entering data for graphs to be used by EZDRAW.
Calls:	EZGO, JCHCLR
Called By:	EZHELP

Name:	DATLNK (DATa LiNK)
Function:	Associates the graph data in the EZDRAW data arrays (INDEPN, DEPEN1, DEPEN2, and DEPEN3) to a particular line/bar/pie graphs. Only data array INDEPN is linked to a pie graph. In line and bar graphs, the user also specifies the independent/dependent axis and provides the start/end points of each axis.
Calls:	EMSGA, EZANSR, EZGO, JAXASC, JCHCLR, JDEPEN, JHAXIS, JINDEP, JRDATA, JVAXIS
Called By:	BARGRF, LINGRF

Name:	DATPRO (DATa PRocessing 0)
Function:	Associates graph data input interactively or from an external data file to data array INDEPN.
Calls:	ENVIRO, EZGO, JCHCLR
Called By:	DATAIN

Name:	DATPR1 (DATa PRocessing 1)
Function:	Associates graph data input interactively or from an external data file to data array DEPEN1.
Calls:	ENVIRO, EZGO, JCHCLR
Called By:	DATAIN

Name:	DATPR2 (DATa PRocessing 2)
Function:	Associates graph data input interactively or from an external data file to data array DEPEN2.
Calls:	ENVIRO, EZGO, JCHCLR
Called By:	DATAIN

Name:	DATPR3 (DATa PRocessing 3)
Function:	Associates graph data input interactively or from an external data file to data array DEPEN3.
Calls:	ENVIRO, EZGO, JCHCLR
Called By:	DATAIN

Name:	DATRA (DATa atTRibutes)
Function:	Provides information about the attributes for data represented in bar, line or pie graphs.
Calls:	EZGO, JCHCLR
Called By:	DSGINF, MODINF

Name:	DESIGN
Function:	Displays the DESIGN Graph Menu and determines which type of graph the user wishes to design.
Calls:	BARGRF, CHKSAV, DMENU, EMSGM, EZHELP, JBAR, JCHART, JCHEXT, JCHPOS, JCHWDW, JDSPOS, JDSWDW, JGRAPH, JPIE, LINGRF, PIEGRF
Called By:	OPTION

Name:	DIFF (DIFFerence)
Function:	Explains the difference between modifying and designing graphs.
Calls:	EZGO, JCHCLR
Called By:	DSGINF, MOD

Name:	DLBL1, DLBL2, DLBL3 (Data LaBeL 1/2/3)
Function:	Allows the user to specify the text and the attributes for the text that will be the label for any of the curves/bars of the respective graph.
Calls:	DSGTXT, EMSGA, EZANSR, JADNOT, JCHCLR, JPONOT, JTXBOX, JTXHGT, JTXJST, JTXPTH, JTXQAL, JTXROT, JXTEXT, POSTXT
Called By:	BARLBL, CURLBL

Name:	DMENU (Design MENU)
Function:	Displays the DESIGN Graph Menu options.
Calls:	JCHCLR
Called By:	DESIGN

Name:	DPORT (Draw [graph] viewPORT)
Function:	Defines that portion of the view space on the display screen where the graph will be drawn/rendered during a design/modify graph session.
Calls:	JVPORT, JWINDO
Called By:	BMENU, LMENU, PMENU

Name:	DSGINF (DeSiGn INFormation)
Function:	Provides information about designing new graphs.
Calls:	EZGO, JCHCLR
Called By:	EZHELP

Name:	DSGTXT (DeSiGn TeXT [strings])
Function:	Allows the user to specify the text string and all its attributes (i.e. quality, path, gap, size, color, ratio, justification, rotation, width, and height).
Calls:	JCHCLR, TXTATR, STRMKR
Called By:	DLBL1, DLBL2, DLBL3, HAXLBL, NOTE1, NOTE2, NOTE3, TITLE1, TITLE2, TITLE3, VAXLBL

Name:	ELMNTS (ELeMeNTS)
Function:	Provides information about the elements of bar, line, and pie graphs.
Calls:	EZGO, JCHCLR
Called By:	DSGINF, MODINF

Name:	EMSGA (Error MeSsaGe "Answer")
Function:	A general purpose message informing the user that an erroneous entry was made to an EZDRAW Prompt for a specific answer.
Calls:	EZGO, JCHCLR
Called By:	CHKHLP, CHKSAV, DATLNK, DLBL1, DLBL2, DLBL3, ENVIRO, GAXES, GAXLBL, GETGRF, HAXLBL, HTIC, LINATR, LINTNS, LSTYLE, LWIDTH, MODIFY, NOTE1, NOTE2, NOTE3, PICSEG, PIELNK, PIPOSN, PLYFIL, POSTXT, PROC1, PROC2, PROC3, PROC4, TXTATR, SAVE, SETGAP, SETSZE, TICLBL, TICMRK, TITLE1, TITLE2, TITLE3, VAXLBL, VIEW, VTIC

Name:	EMSGF (Error MeSsaGe "File")
Function:	Provides an explanation to the user of an error occurring during an attempt to "open" a user specified graph or data file.
Calls:	EZGO, JCHCLR
Called By:	GETGRF, PROC3, PROC4, SAVE

Name:	EMSGM (Error MeSsaGe "Menu")
Function:	Provides an explanation to the user of an erroneous input to an EZDRAW menu prompt.
Calls:	EZGO, JCHCLR
Called By:	BAR, BARGRF, BARLBL, CURLBL, CURVE, DATAIN, DESIGN, EZHELP, LINGRF, LINMRK, LINTNS, LSTYLE, LWIDTH, MODIFY, NOTES, OPTION, PIEGRF, PLYFIL, SETCLR, SETEXT, SETFNT, SETGAP, SETJST, SETLIN, SETPTH, SETQAL, SETROT, TITLE, TYPEAX

Name:	EMSGR (Error MeSsaGe "Record")
Function:	Provides an explanation to the user of an erroneous record number entry to the prompt requesting a user specified record number when saving/retrieving graphs to/from user specified files.
Calls:	EZGO, JCHCLR
Called By:	GETGRF, SAVE

Name:	EMSGRD (Error MeSsaGe "ReaD")
Function:	Provides an explanation to the user of an error occurring during the "reading" of a record from a graph file, or the "writing" of graph data (PSA) into a specified record of a file.
Calls:	EZGO, JCHCLR
Called By:	GETGRF, PROC3, PROC4, SAVE

Name:	ENDEZ (END EZDRAW)
Function:	Terminates EZDRAW, GRAFMAKER, and DI-3000.
Calls:	CHKSAV, JCHTRM, JEND
Called By:	EZDRAW

Name:	ENVIRO ([data input] ENVIROment)
Function:	Determines from the user whether data for the graph will be input interactively or from an external data file.
Calls:	EMSGA, EZANSR, JCHCLR, PROC1, PROC2, PROC3, PROC4
Called By:	DATPR0, DATPR1, DATPR2, DATPR3

Name:	EZANSR (EZDRAW [prompt] ANSweR)
Function:	A utility routine that "reads" the user "answer" to an EZDRAW prompt.
Calls:	None.
Called By:	CHKHLP, CHKSAV, DATLNK, DLBL1, DLBL2, DLBL3, ENVIRO, GAXES, GAXLBL, GETGRF, HAXLBL, LINATR, MODIFY, NOTE1, NOTE2, NOTE3, PIELNK, POSTXT, SAVE, TICLBL, TICMRK, TITLE1, TITLE2, TITLE3, TXTATR, VAXLBL, VIEW

Name:	EZDRAW ("Easy" DRAW)
Function:	Main module.
Calls:	STRTEZ, OPTION, ENDEZ
Called By:	None.

Name:	EZDRIS (EZDRaw IS)
Function:	Provides a general description of EZDRAW.
Calls:	EZGO, JCHCLR
Called By:	GENINF

Name:	EZGO (EZDRAW "GO")
Function:	Acts as a pause and then allows EZDRAW to continue execution.
Calls:	None.
Called By:	BAR1, BAR2, BAR3, CURVE1, CURVE2, CURVE3, DATAIN, DATINF, DATLNK, DATPRO, DATPR1, DATPR2, DATPR3, DSGINF, EMSGA, EMSGF, EMSGM, EMSGR, EMSGRD, EZTITL, HLPINF, HORAX, HTCLBL, HTIC, LINTNS, LSTYLE, LWIDTH, MODINF, PEXPLO, PIELNK, PIPOSN, PISGMT, PLYFIL, PROC1, PROC2, SAVE, SAVINF, SEGLBL, SETRTO, SHOW, VERAX, VTCLBL, VTIC, VUINF

Name:	EZHELP (EZDRAW HELP [Mode])
Function:	Provides the user computer assistance in many of the EZDRAW operations which include: General EZDRAW information, HELP Mode information, Data Entry, Designing Graphs, Modifying Graphs, Viewing Graphs, and Saving Graphs.
Calls:	DATINF, DSGINF, GENINF, HLPINF, HMENU, MODINF, SAVINF, VUINF
Called By:	BAR, BARGRF, BARLBL, CHKHLP, CURLBL, CURVE, DATAIN, DESIGN, LINGRF, LINMRK, LSTYLE, LINTNS, LWIDTH, MODIFY, NOTES, OPTION, PIEGRF, PLYFIL, SETCLR, SETEXT, SETFNT, SETGAP, SETJST, SETLIN, SETPTH, SETQAL, SETROT, TITLE, TYPEAX

Name:	EZTITL (EZDRAW TITLe)
Function:	Prints the EZDRAW "Title Page"
Calls:	EZGO, JADNOT, JCHART, JCHCLR, JCHEXT, JCHSHW, JCLOSE, JMOVE, JOPEN, JPONOT, JTXBOX, JTXJST, JTXPTH, JTXQAL, JTXROT, JUPDAT, JVPORT, JWINDO, JXTEXT
Called By:	STRTEZ

Name:	FATALE (FATAL Errors)
Function:	Refers user to fatal error messages that occur when EZDRAW, GRAFMAKER, and DI-3000 routines terminate abnormally.
Calls:	EZGO, JCHCLR
Called By:	GENINF

Name:	FILINF (FILe INFormation)
Function:	Provides general information on how files are created by EZDRAW.
Calls:	EZGO, JCHCLR
Called By:	SAVINF

Name:	FNAMES (File NAMES)
Function:	Describes file names.
Calls:	EZGO, JCHCLR
Called By:	SAVINF

Name:	FRCDNM (File ReCorD NuMber)
Function:	Describes file record numbers.
Calls:	EZGO, JCHCLR
Called By:	SAVINF

Name:	GAXES (Graph AXES)
Function:	Determines for which axis of a line/bar graph the user wishes to specify line attributes.
Calls:	EMSGA, EXANSR, HORAX, JCHCLR, VERAX
Called By:	BARGRF, LINGRF

Name:	GAXLBL (Graph AXes LaBeL)
Function:	Determines for which axis of a line/bar graph the user wishes to specify a label and its text attributes.
Calls:	EMSGA, EZANSR, HAXLBL, JCHCLR, VAXLBL
Called By:	BARGRF, LINGRF

Name:	GENDAT (GENeral [information on] DATa [entry])
Function:	Provides general information about data entry for data to be used for graphs created with EZDRAW.
Calls:	EZGO, JCHCLR
Called By:	DATINF

Name:	GENDSG (GENeral [information on] DeSiGning)
Function:	Provides user information about designing new graphs.
Calls:	EZGO, JCHCLR
Called By:	DSGINF

Name:	GENINF (GENeral INFormation)
Function:	Provides general information about EZDRAW.
Calls:	EMSGM, EZDRIS, EZGO, FATALE, JCHCLR, NONFTL, SPTSFT
Called By:	EZHELP

Name:	GENMOD (GENeral [information on] MODifying)
Function:	Provides general information on the modification of existing graphs (current and/or saved).
Calls:	EZGO, JCHCLR
Called By:	MODINF

Name:	GENSAV (GENeral [information on] SAVing)
Function:	Provides information about how to save graphs and the save routine.
Calls:	EZGO, JCHCLR
Called By:	SAVINF

Name:	GENVU (GENeral [information on] Viewing)
Function:	Provides information about viewing graphs (current and/or saved).
Calls:	EZGO, JCHCLR
Called By:	VUINF

Name:	GETGRF (GET GRaph)
Function:	Retrieves a user specified graph (by file and record number) of a previously designed and saved graph.
Calls:	BORDER, EMSGA, EMSGF, EMSGR, EMSGRD, EZANSR, JCHCLR, JCLOSE, JJUST, JMARGIN, JMOVE, JSIZE, JOPEN, JUPDAT, JWCLIP, J1TEXT, PPORT, SPORT
Called By:	MODIFY, VIEW

Name:	HAXLBL (Horizontal AXis LaBeL)
Function:	Allows the user to specify the horizontal axis label and its text attributes.
Calls:	DSGTXT, EMSGA, EXANSR, JADNOT, JCHCLR, JPONOT, JTXBOX, JTXHGT, JTXJST, JTXPTH, JTXQAL, JTXROT, JXTEXT, POSTXT
Called By:	GAXLBL

Name: HLPINF (HeLP INFormation)
Function: Provides information about the EZDRAW HELP Mode.
Calls: EMSGM, EZGO, HELPIS, HLPREF, JCHCLR, WHNHLP
Called By: EZHELP

Name: HELPIS (HELP [mode] IS)
Function: Informs user what the EZDRAW HELP Mode is and what it can do for the user during a session.
Calls: EZGO, JCHCLR
Called By: HLPINF

Name: HLPREF ([EZDRAW] HeLP REFerences)
Function: Indicates which manuals are referenced by the HELP Mode.
Calls: EZGO, JCHCLR
Called By: HLPINF

Name: HMENU (Help [mode] MENU)
Function: Displays the EZDRAW HELP Mode Menu options.
Calls: JCHCLR
Called By: EZHELP

Name: HORAX (HORizontal AXis)
Function: Allows the user to specify the horizontal axis line attributes.
Calls: EZGO, JAXATR, JCHCLR, JXLINE, LINATR, TYPEAX
Called By: GAXES

Name: HTCLBL (Horizontal [axis] TiCk [mark] LaBeL)
Function: Allows the user to specify the horizontal axis tick mark label(s) and the text attributes.
Calls: EZGO, JCHCLR, JTCLBL, JTCPAT, JTXBOX, JTXHGT, JTXJST, JTXPTH, JTXQAL, JTXROT, JXTEXT, STRMKR, TXTATR
Called By: TICLBL

Name: HTIC (Horizontal [axis] TICk [mark])
Function: Allows the user to specify the attributes associated with tick marks for the horizontal axis.
Calls: EMSGA, EZGO, JCHCLR, JTCTYP, JTIC, JTCATR, JXLINE, LINATR
Called By: TICMRK

Name: IGAP (Inter-character GAP)
Function: Provides information about the inter-character gap of text strings.
Calls: EZGO, JCHCLR
Called By: TATRA

Name: INITEZ (INITialize EZDRAW)
Function: Initializes EZDRAW data arrays, the GRAFMAKER Picture Storage Area, and sets EZDRAW text and line attribute value to EZDRAW defaults.
Calls: JCHART
Called By: STRTEZ

Name: LATRA (Line ATtRibutes)
Function: Provides information concerning the attributes the user specifies for graphical line elements on all graphs designed with EZDRAW.
Calls: EMSGM, JCHCLR, LNCLR, LNINT, LNSTL, LNWID
Called By: DSGINF, MODINF

Name: LINATR (LINe ATtRibutes)
Function: Allows the user to redefine any of the EZDRAW default values for attributes associated with graphical line elements.
Calls: EMSGA, EZANSR, JCHCLR, SETLIN
Called By: BAR1, BAR2, BAR3, CURVE1, CURVE2, CURVE3, HORAX, HTIC, PISGMT, VERAX, VTIC

Name: LINES
Function: Provides information about line graphs.
Calls: EZGO, JCHCLR
Called By: DSGINF

Name:	LINGRF (LINe GRaph)
Function:	Displays the Line Graph Menu and accesses the EZDRAW routines that allow the user to design and specify the line graph elements.
Calls:	CURLBL, CURVE, DATLNK, EMSGM, EZHELP, GAXES, GAXLBL, JCHCLR, JCHSHW, LMENU, NOTES, TICLBL, TICMRK, TITLE
Called By:	DESIGN, MODIFY

Name:	LINMRK (LINe MARKer)
Function:	Allows the user to specify which type of marker is to be used along a curve to identify the data points of a particular curve.
Calls:	EMSGM, EZHELP, JCHCLR
Called By:	CURVE1, CURVE2, CURVE3

Name:	LINTNS (Line INTeNSity)
Function:	Allows the user to specify the line attribute intensity/brightness for all line graph elements.
Calls:	EMSGA, EMSGM, EZGO, EZHELP, JCHCLR
Called By:	SETLIN

Name:	LMENU (Line [graph] MENU)
Function:	Displays the Line Graph Menu Options available to the user during design/modify sessions.
Calls:	AREALM, BORDER, DPORT, JWCLIP, MPORT, PPORT, PROMPT
Called By:	LINGRF

Name:	LNCLR (LiNe CoLoR)
Function:	Provides information about the color of linear graph elements on all graphs.
Calls:	EZGO, JCHCLR
Called By:	LATRA

Name:	LNINT (LiNe INTensity)
Function:	Provides information about the intensity of linear graph elements on all graphs.
Calls:	EZGO, JCHCLR
Called By:	LATRA

Name:	LNKDAT (LiNK DATa [to line and bar graphs])
Function:	Describes the linking procedure of EZDRAW data arrays to particular line/bar graphs being designed.
Calls:	EZGO, JCHCLR
Called By:	DATINF

Name:	LNKPIE (LiNK PIE)
Function:	Provides information about what occurs when the EZDRAW independent variable data array (INDEPN) is linked to a pie graph being designed.
Calls:	EZGO, JCHCLR
Called By:	DATINF

Name:	LNSTL (LiNe STyLe)
Function:	Provides information about the line styles of linear graph elements in all graphs.
Calls:	EZGO, JCHCLR
Called By:	LATRA

Name:	LNWID (LiNe WIDth)
Function:	Provides information about the width of linear graph elements in all graphs.
Calls:	EZGO, JCHCLR
Called By:	LATRA

Name:	LSTYLE (Line [element] STYLE)
Function:	Allows the user to specify the attribute line style for graph line elements (i.e. whether the line will be solid, dashed, etc.).
Calls:	EMSGA, EMSGM, EZGO, EZHELP, JCHCLR
Called By:	SETLIN

Name:	LWIDTH (Line [element] WIDTH)
Function:	Allows the user to specify the attribute line width for graph line elements (i.e. thickness of the line element).
Calls:	EMSGA, EMSGM, EZGO, EZHELP, JCHCLR
Called By:	SETLIN

Name: METFIL (METa-FILe)
Function: Describes the DI-3000 METAFILE system.
Calls: EZGO, JCHCLR
Called By: SAVINF

Name: MODIFY
Function: Allows the user to change a previously designed graph.
Calls: BARGRF, CHKHLP, EMSGA, EMSGM, EZANSR, EZHELP, GETGRF, JCHART, JCHCLR, LINGRF, PIEGRF
Called By: OPTION

Name: MODINF (MODify INFormation)
Function: Provides information about modifying graphs (current and/or saved).
Calls: DATRA, DIFF, EMSGM, ELMNTS, EZGO, GENMOD, JCHCLR, LATRA, RECALL, TATRA
Called By: EZHELP

Name: MPORT (Menu viewPORT)
Function: Defines that portion of the view space on the display screen where the Line/Bar/Pie Graph Menu will be displayed during a design/modify graph session.
Calls: JVPORT, JWINDO
Called By: BMENU, LMENU, PMENU

Name: NEWFIL (NEW FILe)
Function: Describes EZDRAW prompts to new graph files.
Calls: EZGO, JCHCLR
Called By: SAVINF

Name: NONFTL (NON-FaTaL [errors])
Function: Describes EZDRAW non-fatal error messages.
Calls: EZGO, JCHCLR
Called By: GENINF

Name: NOTES
Function: Determines for which of the three additional graph notations the user wishes to specify text and attributes.
Calls: EMSGM, EZHELP, JCHCLR, NOTE1, NOTE2, NOTE3
Called By: BARGRF, LINGRF, PIEGRF

Name:	NOTE1, NOTE2, NOTE3
Function:	Allows the user to specify the text and the text attributes for the particular additional graph notations EZDRAW allows per graph.
Calls:	DSGTXT, EMSGA, EZANSR, JADNOT, JCHCLR, JPOINT, JTXBOX, JTXHGT, JTXJST, JTXPTH, JTXQAL, JTXROT, JXTEXT, POSTXT
Called By:	NOTES

Name:	OPTION
Function:	Displays the EZDRAW Command Menu and determines which of the EZDRAW options the user wishes to enter. This is the main driver module of EZDRAW.
Calls:	CDMENU, DATAIN, DESIGN, EMSGM, EZHELP, MODIFY, SAVE, VIEW
Called By:	EZDRAW

Name:	PEXPLO (Pie EXPLOde)
Function:	Allows the user to specify which contiguous segments of a pie graph will be "exploded"/ detached from the pie by a fixed percentage (0.20) of the pie graph radius.
Calls:	EZGO, JCHCLR, JSGEXP, PICSEG
Called By:	PIEGRF

Name:	PICSEG (PICk [pie] SEGment)
Function:	Determines which particular segment of a pie graph the user wishes to "process" in some graphical manner (i.e. specify segment attribute, specify segment text label, explode, etc.).
Calls:	EMSGA, JCHCLR
Called By:	PEXPLO, PISGMT, SEGLBL

Name:	PIDAT (PIe [graph] DATa)
Function:	Provides information about the pie segment data of pie graphs.
Calls:	EZGO, JCHCLR
Called By:	DATRA

Name:	PIEGRF (PIE GRaph)
Function:	Displays the Pie Graph Menu and accesses all the routines that allow the user to design the elements of a pie graph.
Calls:	EMSGM, EZHELP, JCHCLR, JCHSHW, NOTES, PEXPLO, PIELNK, PIPOSN, PISGMT, PMENU, SEGLBL, TITLE
Called By:	DESIGN, MODIFY

Name:	PIELNK (PIE [graph data] LiNK)
Function:	Allows the user to associate the data in the data array INDEPN to the pie graph to be used as the segments of the pie graph.
Calls:	EMSGA, EZANSR, EZGO, JCHCLR, JPSEGS, JRDATA
Called By:	PIEGRF

Name:	PIES
Function:	Provides general information about pie graphs.
Calls:	EZGO, JCHCLR
Called By:	DSGINF

Name:	PIPOSN (PIe [graph] POSitioN)
Function:	Allows the user to redefine the position of the pie graph (in chart coordinates), the radius of the pie graph, the direction in which the segments will be drawn, and a rotation from the horizontal to where the first segment will be drawn.
Calls:	EMSGA, EZGO, JCHCLR, JPIATR
Called By:	PIEGRF

Name:	PISGMT (PIe [graph] SeGMenT)
Function:	Allows the user to specify the attributes for a particular segment of a pie graph (i.e. the line attributes for the outline of the segment, the color/interior fill pattern, etc.).
Calls:	EZGO, JCHCLR, JSGATR, JXLINE, LINATR, PICSEG, PLYFIL, SETCLR
Called By:	PIEGRF

Name:	PLYFIL (PoLYgon [interior] FILl [pattern])
Function:	Allows the user to specify the interior fill pattern for graph polygons (bars and pie segments).
Calls:	EMSGA, EMSGM, EZGO, EZHELP, JCHCLR
Called By:	BAR1, BAR2, BAR3, PISGMT

Name:	PMENU (Pie [graph] MENU)
Function:	Displays the user options, in menu format, for specifying the elements of pie graph during a modify/design graph session.
Calls:	AREAPM, BORDER, DPORT, JWCLIP, MPORT, PPORT, PROMPT
Called By:	PIEGRF

Name:	POSTXT (POSition [graph] TeXT [strings])
Function:	Allows the user to redefine the position of graph text strings (in chart coordinates) relative to the justification point of the string.
Calls:	EMSGA, EZANSR, JCHCLR
Called By:	DLBL1, DLBL2, DLBL3, HAXLBL, NOTE1, NOTE2, NOTE3, TITLE1, TITLE2, TITLE3, VAXLBL

Name:	PPORT (Prompt viewPORT)
Function:	Defines that portion of the view space on the display screen where the graph menu prompt will be displayed during a design/ modify graph session and when EZDRAW is retrieving another graph for modifying/ viewing.
Calls:	JVPORT, JWINDO
Called By:	BMENU, GETGRF, LMENU, PMENU

Name:	PROC1, PROC2 ([data] PROCessing 1/2)
Function:	Allows the user to input data for graphs interactively. Real data is input with PROC1 while integer data is input with PROC2. Integer data is converted to real.
Calls:	EMSGA, EZGO, JCHCLR
Called By:	ENVIRO

Name:	PROC3, PROC4 ([data] PROCessing 3/4)
Function:	Allows the user to input data for graphs from external data files. Real data is input with PROC3 while integer data is input with PROC4. Integer data is convert i to real.
Calls:	EMSGA, EMSGF, EMSGRD, JCHCLR
Called By:	ENVIRO

Name:	PROMPT
Function:	Displays a prompt to the user to enter an option from the bar/line/pie graph menu in the "prompt viewport".
Calls:	JCLOSE, JJUST, JMARGN, JMOVE, JOPEN, JSIZE, JUPDAT, J1TEXT
Called By:	BMENU, LMENU, PMENU

Name:	RECALL
Function:	Describes procedures for recalling a previously designed graph saved in a graph file.
Calls:	EZGO, JCHCLR
Called By:	MODINF

Name:	RETREV (RETRiEVe)
Function:	Explains the procedures for retrieving a previously designed and saved graph for viewing/taking pictures.
Calls:	EZGO, JCHCLR
Called By:	VUINF

Name:	SAVE
Function:	Allows the user to "save" the graph in the GRAFMAKER Picture Storage Area by writing the contents of the PSA to direct access file. The name of the file and the record in the file in which the graph is written, is specified by the user. If the file is new, EZDRAW will create it for the user. If the file already exists, EZDRAW will write the graph to an already existing record or a new record. The graphs may be retrieved later by the user. The user also has the option of saving the graph in the DI-3000 Metafile.
Calls:	BIGPRT, BORDER, CHKHLP, EMSGA, EMSGF, EMSGR, EMSGRD, EZANSR, EZGO, JCHCLR, JCHSHW, JDSUNT, JSUNIT, JUPDAT
Called By:	CHKSAV, OPTION

Name:	SAVINF (SAVe INFormation)
Function:	Provides information about the means to save the graphical data in the picture storage area in an external graph file.
Calls:	CAUTNS, EMSGM, EZGO, FILINF, FNAMES, FRCDNM, GENSAV, JCHCLR, METFIL, NEWFIL
Called By:	EZHELP

Name:	SEGLBL ([pie graph] SEGment LaBeL)
Function:	Allows the user to specify the text label and the text attributes for a particular segment of a pie graph. The user also has the option of displaying the segment label, percentage, relative value, or a combination of the three. All labels are positioned by GRAFMAKER based on the position of the segment. EZDRAW places all labels outside of the segment.
Calls:	EZGO, JCHCLR, JSGFMT, JSLFMT, JSGLBL, JTXBOX, JTXPTH, JTXQAL, JXTEXT, PICSEG, STRMKR, TXTATR
Called By:	PIEGRF

Name:	SETCLR (SET CoLoR [attribute])
Function:	Allows the user to specify the color attribute for the graph element. Also redefines the color EZDRAW default value.
Calls:	EMSGM, EZHELP, JCHCLR
Called By:	BAR1, BAR2, BAR3, PISGMT, SETEXT, SETLIN

Name:	SETEXT (SEt TEXT [attributes])
Function:	Allows the user to specify all the attributes associated with graphical text elements. Also redefines the EZDRAW default value for the attribute specified. These attributes include: color, quality, path, gap, font, horizontal and vertical justification, rotation, character width and height, and character width to height ratio.
Calls:	EMSGM, EZHELP, JCHCLR, SETCLR, SETFNT, SETGAP, SETJST, SETPTH, SETQAL, SETROT, SETRTO, SETSZE
Called By:	TXTAIR

Name:	SETFNT (SET [text] FoNT [attribute])
Function:	Allows the user to specify the font attribute for the graphical text element. Text font is the "print/lettering style" of the characters. Also redefines the EZDRAW default font value.
Calls:	EMSGM, EZHELP, JCHCLR
Called By:	SETEXT

Name:	SETGAP (SET [text inter-character] GAP)
Function:	Allows the user to specify the inter-character gap attribute for the graphical text element. Text gap is based on a "text box" which surrounds each character. The size of the "text box" is determined by the width and height of the character. EZDRAW sets the vertical and horizontal "text box" values to 0.02 times the height of the character in chart units.
Calls:	EMSGA, EMSGM, EZHELP, JCHCLR
Called By:	SETEXT

Name:	SETJST (SET [text] JuSTification [attribute])
Function:	Allows the user to specify the horizontal and vertical justification point of a graphical text element. Also redefines the justification EZDRAW default values.
Calls:	EMSGM, EZHELP, JCHCLR
Called By:	SETEXT

Name: SETLIN (SET LINe [attributes])
Function: Allows the user to specify all the attributes associated with graphical line elements. Also redefines the EZDRAW default values for those attributes specified.
Calls: EMSGM, EZHELP, JCHCLR, LINTNS, LSTYLE, LWIDTH, SETCLR
Called By: LINATR

Name: SETPTH (SET [text] PaTH [attribute])
Function: Allows the user to specify the path attribute for graphical text elements. Also redefines the EZDRAW text path default value.
Calls: EMSGM, EZHELP, JCHCLR
Called By: SETEXT

Name: SETQAL (SET [text] QuALity [attribute])
Function: Allows the user to specify the quality attribute for graphical text elements. Also redefines the EZDRAW text quality default value.
Calls: EMSGM, EZHELP, JCHCLR
Called By: SETEXT

Name: SETROT (SET [text] ROTation [angle])
Function: Allows the user to specify the rotation angle attribute for graphical text elements. Also redefines the text rotation angle EZDRAW default value.
Calls: EMSGA, EMSGM, EZHELP, JCHCLR
Called By: SETEXT

Name: SETRTO (SET [character width to height] RaTiO)
Function: Currently informs the user that the character width to height ratio of EZDRAW text strings is a system standard set to 0.75. The user is not currently able to redefine this value.
Calls: EZGO, JCHCLR
Called By: SETEXT

Name:	SETSZE (SET [character] SiZE)
Function:	Allows the user to specify the character size attribute for graphical text elements. Also redefines the text size EZDRAW default value.
Calls:	EMSGA, JCHCLR
Called By:	SETEXT

Name:	SHOW
Function:	Displays the current graph or a graph retrieved from a graph file enlarged (utilizing the maximum view space of the display screen) in order for the user to view the graph and/or make a hard copy of the graph on an external "graphics copier" (4007 Color Graphics Camera if the user is on RAMTEK 6212 or the 4361 Hard Copy Unit if the user is on the TEKTRONIX 4014/4016).
Calls:	BIGPRT, BORDER, EZGO, JCHCLR, JCHSHW, JUPDAT, JWCLIP
Called By:	VIEW

Name:	SPORT (Show viewPORT)
Function:	Defines the viewport and window for displaying graphs that are retrieved from graph files to determine if the correct graph was retrieved.
Calls:	BORDER, JCHSHW, JVPORT, JWINDO
Called By:	GETGRF

Name:	SPTSFT (SuPporT SoFTware)
Function:	Describes the Precision Visuals graphics software supporting EZDRAW.
Calls:	EZGO, JCHCLR
Called By:	GENINF

Name:	STRMKR ([EZDRAW text] STRing MaKeR)
Function:	Takes the input text string (a FORTRAN 77 character variable) and adds the EZDRAW string delimiter "^" to the front and end (immediately behind the last non-blank character). Then the string is converted to an equivalent "integer text string" which is compatible for GRAFMAKER.
Calls:	None.
Called By:	DSGTXT, HTCLBL, SEGLBL, VTCLBL

Name: STRTEZ (STaRT EZDRAW)
Function: Initiates DI-3000 and GRAFMAKER. Initializes the EZDRAW data arrays, sets attribute values for graphical line and text elements to EZDRAW defaults, prints the EZDRAW title page, sets the flags, and then determines if the user wishes to enter the EZDRAW HELP Mode.
Calls: CHKHLP, EZTITL, INITEZ, JBEGIN, JCHINI, JDEVON, JDINIT, JVSPAC
Called By: EZDRAW

Name: TAKPIX (TAKe PIcture [of graph])
Function: Describes how to make hard copies of the graphics output.
Calls: EZGO, JCHCLR
Called By: VUINF

Name: TATRA (Text ATtRibutes)
Function: Provides information concerning the attributes associated with graphical text elements that the user specifies.
Calls: EMSGM, IGAP, JCHCLR, TCLR, TFNT, TJST, TPTH, TQLTY, TROTN, TRTIO, TSIZE
Called By: DSGINF, MODINF

Name: TCLR (Text CoLoR)
Function: Provides information about the color of text strings for all graphs.
Calls: EZGO, JCHCLR
Called By: TATRA

Name: TFNT (Text FoNT)
Function: Provides information about the type face/ print style of text strings for all graphs.
Calls: EZGO, JCHCLR
Called By: TATRA

Name: TICLBL (TICk [mark] LaBeL)
Function: Determines for which axis tick marks, the user wishes to specify the tick mark labels, and the attributes for the tick mark labels.
Calls: EMSGA, EZANSR, HTCLBL, JCHCLR, VTCLBL
Called By: BARGRF, LINGRF

Name:	TICMRK (TICk MaRKs)
Function:	Allows the user to specify the tick mark labels and their attributes for tick marks along the horizontal or vertical axes of line or bar graphs. The user designates which tick marks along the axis will be labelled.
Calls:	EMSGA, EZANSR, HTIC, JCHCLR, VTIC
Called By:	BARGRF, LINGRF

Name:	TITLE
Function:	Determines for which of three graph titles the user wishes to specify text and attributes.
Calls:	EMSGM, EZHELP, JCHCLR, TITLE1, TITLE2, TITLE3
Called By:	BARGRF, LINGRF, PIEGRF

Name:	TITLE1, TITLE2, TITLE3
Function:	Allows the user to specify text and the text attributes for any of the three titles allowed by EZDRAW for line/bar/pie graphs.
Calls:	DSGTXT, EZANSR, JADNOT, JCHCLR, JPONOT, JTXBOX, JTXHGT, JTXJST, JTXPTH, JTXQAL, JTXROT, JXTEXT, POSTXT
Called By:	TITLE

Name:	TJST (Text JuSTification)
Function:	Provides information about the vertical and horizontal justification point of text strings for all graphs.
Calls:	EZGO, JCHCLR
Called By:	TATRA

Name:	TPTH (Text PaTH)
Function:	Provides information about the direction in which text strings are written for all graphs.
Calls:	EZGO, JCHCLR
Called By:	TATRA

Name:	TQLTY (Text QuaLiTY)
Function:	Provides information about the quality/ precision with which text strings follow the other text attributes.
Calls:	EZGO, JCHCLR
Called By:	TATRA

Name: TROTN (Text ROTatioN)
Function: Provides information about rotating text strings for all graphs.
Calls: EZGO, JCHCLR
Called By: TATRA

Name: TRTIO (Text RaTIO)
Function: Describes the text character width to height ratio for all text strings.
Calls: EZGO, JCHCLR
Called By: TATRA

Name: TSZE (Text SiZE)
Function: Describes the manner in which the size of text characters are specified for text strings.
Calls: EZGO, JCHCLR
Called By: TATRA

Name: TXTATR (TeXT ATtRibutes)
Function: Allows the user to either apply the current EZDRAW text attribute default values for a text string or to selectively redefine the attribute value which then becomes the new current EZDRAW text attribute default.
Calls: EMSGA, EZANSR, JCHCLR, SETEXT
Called By: DSGTXT, HTCLBL, VTCLBL

Name: TYPEAX (TYPE AXis)
Function: Allows the user to specify the type of axis to be used for the horizontal or vertical axis of line and bar graphs.
Calls: EMSGM, EZHELP, JCHCLR
Called By: HORAX, VERAX

Name: VAXLBL (Vertical AXis LaBeL)
Function: Allows the user to specify the text and text attributes for the vertical axis label for bar and line graphs.
Calls: DSGTXT, EMSGA, EZANSR, JADNOT, JCHCLR, JPONOT, JTXBOX, JTXHGT, JTXJST, JTXPTH, JTXQAL, JTXROT, JXTEXT, POSTXT
Called By: GAXLBL

Name:	VERAX (VERtical AXis)
Function:	Allows the user to specify the line and axis attributes for the vertical axis for line and bar graphs.
Calls:	EZGO, JAXATR, JCHCLR, JXLINE, LINATR, TYPEAX
Called By:	GAXES

Name:	VIEW
Function:	Allows the user to "view" and/or make a hard copy of the current graph or another graph retrieved from a graph file. The graph is enlarged and utilizes the maximum view space of the display screen. If the user is on the RAMTEK 6212, the 4007 Color Graphics Camera may be used for copying graphics output. The user may use the 4631 Hard Copy Unit if on the TEKTRONIX 4014/4016.
Calls:	CHKHLP, CHKSAV, EMSGA, EZNASR, GETGRF, JCHART JCHCLR, SHOW
Called By:	OPTION

Name:	VTCLBL (Vertical [axis] TiCk [mark] LaBeL)
Function:	Allows the user to specify the tick mark labels and text attributes for tick marks along the vertical axis of line and bar graphs. The user will specify which tick mark along the vertical axis will be labelled.
Calls:	EZGO, JCHCLR, JTCLBL, JTCPAT, JTXBOX, JTXHGT, JTXJST, JTXPTH, JTXQAL, JTXROT, JXTEXT, STRMKR, TXTATR
Called By:	TICLBL

Name:	VTIC (Vertical [axis] TICk [marks])
Function:	Allows the user to specify the line and tick mark attributes for tick marks along the vertical axis for line and bar graphs.
Calls:	EMSGA, EZGO, JCHCLR, JTCATR, JTCTYP, JTIC, JXLINE, LINATR
Called By:	TICMRK

Name: VUINF (Viewing [graphs] INFormation)
Function: Provides information about how graphs are viewed (current and/or saved).
Calls: EMSGM, EZGO, GENVU, JCHCLR, RETREV, TAKPIX
Called By: EZHELP

Name: WHNHLP (WHeN HeLP)
Function: Informs user when the EZDRAW HELP Mode is accessible.
Calls: EZGO, JCHCLR
Called By: HLPINF

Name: XITDSG (eXIT DeSiGn)
Function: Provides information about the status of a graph when the design mode is exited.
Calls: EZGO, JCHCLR
Called By: DSGINF

Appendix E

DI-3000 Subroutines Called by EZDRAW

TABLE IV - DI-3000 Subroutines

Name:	JBEGIN
Function:	To initialize DI-3000.
Called By:	STRTEZ
Ref:	A-8
Name:	JCLOSE
Function:	To close the currently open temporary segment.
Called By:	AREABM, AREALM, AREABM, BORDER, EZTITL, GETGRF, PROMPT, SAVE
Ref:	A-15
Name:	JCRLF
Function:	To cause a carriage return and line feed relative to the current margin point. The current position is set to the new current margin point.
Called By:	AREABM, AREALM, AREAPM
Ref:	A-25
Name:	JDEVON/JDINIT
Function:	To initialize a display device.
Called By:	STRTEZ
Ref:	A-29
Name:	JEND
Function:	To terminate DI-3000.
Called By:	ENDEZ
Ref:	A-8
Name:	JIWIND
Function:	To return the 3D, untransformed world coordinates of the viewplane clipping window.
Called By:	BORDER
Ref:	A-64

Name:	JJUST
Function:	To set the current character string justification primitive attribute.
Called By:	AREABM, AREALM, AREAPM, EZTITL, GETGRF, PROMPT
Ref:	A-66

Name:	JMARGN
Function:	To set the world coordinate character margin for subsequent calls to JCRLF within the open segment.
Called By:	AREABM, AREALM, AREAPM, GETGRF, PROMPT
Ref:	A-75

Name:	JMOVE
Function:	To move invisibly from the current position to the specified world coordinate position.
Called By:	AREABM, AREALM, AREAPM, BORDER, EZTITL, GETGRF, PROMPT
Ref:	A-80

Name:	JOPEN
Function:	To open a temporary segment.
Called By:	AREABM, AREALM, AREAPM, BORDER, EZTITL, GETGRF, PROMPT, SAVE
Ref:	A-82

Name:	JPOLY
Function:	To define a connected sequence of visible lines in world coordinates.
Called By:	BORDER
Ref:	A-107

Name:	JSIZE
Function:	To set the dimensions of the current character box.
Called By:	AREABM, AREALM, AREAPM, EZTITL, GETGRF, PROMPT
Ref:	A-126

Name: JUPDAT
Function: To flush any internal DI-3000 buffers, guaranteeing that the image on all selected display devices is current. Also allows FORTRAN 77 input/output.
Called By: EZTITL, GETGRF, PROMPT, SAVE, SHOW
Ref: A-135

Name: JVPORT
Function: To define the boundaries of the virtual coordinate system viewport.
Called By: BIGPRT, DPORT, EZTITL, MPORT, PPORT, SPORT
Ref: A-142

Name: JVSPAC
Function: To redefine the dimensions of the virtual coordinate system and the mapping from the virtual coordinate system onto the visible region of all selected display devices.
Called By: STRTEZ
Ref: A-145

Name: JWCLIP
Function: To control the viewplane clipping function.
Called By: BMENU, GETGRF, LMENU, PMENU, SHOW
Ref: A-149

Name: JWINDO
Function: To define the boundaries of the viewplane clipping window in the UVN viewing coordinate system.
Called By: BIGPRT, DPORT, EZTITL, MPORT, PPORT, SPORT
Ref: 62-63 (UVN Coordinate System), A-150 (JWINDO)

Name: J1TEXT
Function: To output a low-quality (string-precision) text string as an output primitive.
Called By: AREABM, AREALM, AREAPM, EZTITL, GETGRF, PROMPT
Ref: A-153

Appendix F

GRAFMAKER Subroutines Called by EZDRAW

TABLE V - GRAFMAKER Subroutines

Name:	JADNOT
Function:	To add annotation to a chart.
Called By:	DLBL1, DLBL2, DLBL3, EZTITL, HAXLBL, NOTE1, NOTE2, NOTE3, TITLE1, TITLE2, TITLE3, VAXLBL
Ref:	A-2
Name:	JAXASC
Function:	To associate a given curve with an existing axis.
Called By:	DATLNK
Ref:	A-4
Name:	JAXATR
Function:	To provide line, scale factor, and scale type attributes for an axis.
Called By:	HORAX, VERAX
Ref:	A-6
Name:	JBAR
Function:	To initiate the creation of a new bar graph.
Called By:	DESIGN
Ref:	A-9
Name:	JBRATR
Function:	To specify the width and position of the bars of one dependent data set on a bar graph.
Called By:	BAR1, BAR2, BAR3
Ref:	A-10
Name:	JCHART
Function:	To initialize a GRAFMAKER picture storage area.
Called By:	DESIGN, EZTITL, INITEZ, MODIFY, VIEW
Ref:	A-18

Name: JCHCLR
Function: To clear the view surface on all selected devices.
Called By: All EZDRAW subroutines except - DESIGN, ENDEZ, EZDRAW, INITEZ, OPTION, STRTEZ
Ref: A-20

Name: JCHEXT
Function: To define explicitly the X-extent and Y-extent for the chart/picture coordinate space.
Called By: DESIGN, EZTITL
Ref: A-22

Name: JCHINI
Function: To initialize GRAFMAKER.
Called By: STRTEZ
Ref: A-24

Name: JCHPOS
Function: To position a chart within the picture space.
Called By: DESIGN
Ref: A-26

Name: JCHSHW
Function: To display a GRAFMAKER picture on all selected display devices.
Called By: BARGRF, EZTITL, LINGRF, PIEGRF, SAVE, SHOW, SPORT
Ref: A-27

Name: JCHTRM
Function: To terminate GRAFMAKER.
Called By: ENDEZ
Ref: A-28

Name: JCHWDW
Function: To define the region of the picture coordinate system into which the chart space of a specific graph will be mapped (only one graph per picture in EZDRAW).
Called By: DESIGN
Ref: A-29

Name: JDEPEN
Function: To specify a previously defined array of data values to be used as a dependent variable of a line or bar graph.
Called By: DATLNK
Ref: A-35

Name: JDSPOS
Function: To define the position of the data space within a line or bar graph.
Called By: DESIGN
Ref: A-37

Name: JDSUNT
Function: To deselect a display device for GRAFMAKER output.
Called By: SAVE
Ref: A-38

Name: JDSWDW
Function: To define the size of the data space for a line or bar graph.
Called By: DESIGN
Ref: A-39

Name: JDTATR
Function: To set the line attributes of a curve on a line graph or the border attributes of bars in a bar graph.
Called By: BAR1, BAR2, BAR3, CURVE1, CURVE2, CURVE3
Ref: A-40

Name: JDTFIL
Function: To set the color and pattern used to fill the bars on a bar graph.
Called By: BAR1, BAR2, BAR3
Ref: A-42

Name: JGRAPH
Function: To initiate the creation of a new line graph.
Called By: DESIGN
Ref: A-47

Name: JHAXIS
Function: To define a horizontal axis for a line or bar graph.
Called By: DATLNK
Ref: A-48

Name: JINDEP
Function: To specify a previously defined array of data values to be used as the independent variable of a line or bar graph.
Called By: DATLNK, PIELNK
Ref: A-50

Name: JPIATR
Function: To define the center point, radius, starting angle, and direction in which data values are rendered for a pie graph.
Called By: PIPOSN
Ref: A-56

Name: JPIE
Function: To initiate the creation of a new pie graph.
Called By: DESIGN
Ref: A-57

Name: JPONOT
Function: To specify a non-default position for an annotation string.
Called By: DLBL1, DLBL2, DLBL3, EZTITL, HAXLBL, NOTE1, NOTE2, NOTE3, TITLE1, TITLE2, TITLE3, VAXLBL
Ref: A-58

Name: JPSEGS
Function: To associate a data set defined in a previous call to JINDEP with the segments in a pie graph.
Called By: PIELNK
Ref: A-59

Name: JRDATA
Function: To define an array of REAL values as a data set
Called By: DATLNK, PIELNK
Ref: A-60

Name: JSGATR
Function: To define the interior color and pattern attributes and the edge style attribute of a single pie graph segment.
Called By: PISGMT
Ref: A-61

Name: JSGEXP
Function: To define a group of contiguous pie graph segments to be "exploded".
Called By: PEXPLO
Ref: A-62

Name: JSGFMT
Function: To define the location (inside/outside) of the pie segment quantity/percentage label for a single pie graph segment.
Called By: SEGLBL
Ref: A-63

Name: JSGLBL
Function: To define a text label for a pie graph segment.
Called By: SEGLBL
Ref: A-64

Name: JSLFMT
Function: To define the FORTRAN FORMAT statements to be used in displaying the pie segment quantities and percentages for all segments in the pie graph.
Called By: SEGLBL
Ref: A-66

Name: JSUNIT
Function: To initialize and select a view surface for GRAFMAKER output.
Called By: SAVE
Ref: A-68

Name: JTCATR
Function: To specify length and line attributes for all tick marks in a group.
Called By: HTIC, VTIC
Ref: A-69

Name: JTCLBL
Function: To define one or more alphanumeric tick mark labels along a specific axis.
Called By: HTCLBL, VTCLBL
Ref: A-73

Name: JTCPAT
Function: To indicate which tick marks in a given tick mark group will be labelled.
Called By: HTCLBL, VTCLBL
Ref: A-75

Name: JTCTYP
Function: To specify the type of a tick mark group and its line style.
Called By: HTIC, VTIC
Ref: A-77

Name: JTIC
Function: To define a group of tick marks along a specific axis.
Called By: HTIC, VTIC
Ref: A-78

Name: JTXBOX
Function: To set the current text box and other attributes for subsequently defined text strings.
Called By: DLBL1, DLBL2, DLBL3, EZTITL, HAXLBL, HTCLBL, NOTE1, NOTE2, NOTE3, SEGLBL, TITLE1, TITLE2, TITLE3, VAXLBL, VTCLBL
Ref: A-80

Name: JTXHGT
Function: To set the current character height text attribute for subsequently defined text strings. The dimensions of the box that surround strings are defined relative to the character height.
Called By: DLBL1, DLBL2, DLBL3, HAXLBL, HTCLBL, NOTE1, NOTE2, NOTE3, SEGLBL, TITLE1, TITLE2, TITLE3, VAXLBL, VTCLBL
Ref: A-82

Name: JTXJST
Function: To set the current horizontal and vertical justification for subsequently defined text strings.
Called By: DLBL1, DLBL2, DLBL3, EZTITL, HAXLBL, HTCLBL, NOTE1, NOTE2, NOTE3, TITLE1, TITLE2, TITLE3, VTCLBL
Ref: A-83

Name: JTXPTH
Function: To set the current path attribute for subsequently defined text strings.
Called By: DLBL1, DLBL2, DLBL3, EZTITL, HAXLBL, HTCLBL, NOTE1, NOTE2, NOTE3, SEGLBL, TITLE1, TITLE2, TITLE3, VAXLBL, VTCLBL
Ref: A-84

Name: JTXQAL
Function: To set the current text quality for subsequently defined text strings.
Called By: DLBL1, DLBL2, DLBL3, EZTITL, HAXLBL, HTCLBL, NOTE1, NOTE2, NOTE3, SEGLBL, TITLE1, TITLE2, TITLE3, VAXLBL, VTCLBL
Ref: A-85

Name: JTXROT
Function: To set the current angle at which subsequently defined text strings will be drawn.
Called By: DLBL1, DLBL2, DLBL3, EZTITL, HAXLBL, HTCLBL, NOTE1, NOTE2, NOTE3, SEGLBL, TITLE1, TITLE2, TITLE3, VAXLBL, VTCLBL
Ref: A-87

Name: JVAXIS
Function: To define a vertical axis for a line or bar graph.
Called By: DATLNK
Ref: A-90

Name: JXLINE
Function: To define a line attribute index representing a combination of color, intensity, line style, and line width.
Called By: BAR1, BAR2, BAR3, CURVE1, CURVE2, CURVE3, HORAX, HTIC, PISGMT, VERAX, VTIC
Ref: A-94

Name: JXTEXT
Function: To define a text attribute index representing a combination of text attributes, including font, gap, character aspect ratio, and color.
Called By: DLBL1, DLBL2, DLBL3, EZTITL, HAXLBL, HTCLBL, NOTE1, NOTE2, NOTE3, SEGLBL, TITLE1, TITLE2, TITLE3, VAXLBL, VTCLBL
Ref: A-95

Appendix G

Implementing the Dynamic Modification Feature

This appendix outlines a possible solution to EZDRAW's incapability to dynamically modify already specified graph elements. The solution is offered since GRAFMAKER does not allow the picture data structure to be altered during execution of a GRAFMAKER application program such as EZDRAW. Also, the possibility to modify graph elements is an existing option since EZDRAW was designed for this dynamic modification capability.

The Problem

Currently, if an EZDRAW user is designing or has designed a graph and wants to change a graph element that has already been specified, then a new graph must be created. The user can also start the redesign from a copy of the graph in which the element to be changed does not appear (i.e. the parameters for the element have not been passed to GRAFMAKER and integrated into the picture data structure).

During a graph design session, the user selects the graph element (data, line, or text) to be defined from the graph menu of the graph type being designed or modified. Once the particular graph element has been selected, interactive routines are called and entered. These routines allow the user to specify all the attributes for the graph element.

These user specified attributes are then passed to GRAFMAKER as arguments in the parameter list of the GRAFMAKER subroutine call. At this point, the GRAFMAKER picture data structure manager takes the values of these attributes and places them in the picture data structure associated with the graph. EZDRAW then invokes the GRAFMAKER rendering routines which display the graph to the screen. The graph menu is then displayed allowing the user to select another graph element to be designed.

If the user now attempts to redefine the parameters in the picture data structure for the graph element just designed, EZDRAW determines if the element has already been designed. Since it has in this case, a non-fatal error message indicates that the user may specify a graph element only once. The user is then returned to the appropriate graph menu.

A Possible Solution

The proposed scheme consists of two components. The first component is a new picture data structure managed by EZDRAW. For the sake of distinction from the GRAFMAKER picture data structure, this new data structure will be called the Graph Data Structure (GDS). The second component is a module that scans the GDS (Subroutine SCAN_GDS) and passes the values of the graph element parameters to GRAFMAKER. These two components will be described later.

The sequence for this proposed system is unchanged from

the viewpoint of the user. From the graph menu, the interactive routines allowing the user to specify the attributes for the graph element are entered. Once all the values for the attributes are provided by the user, EZDRAW places these values into the area of the GDS reserved for the particular graph element. These graph element areas will be called fields and are described later. The interactive routines are then exited.

At this point, the GRAFMAKER picture data area is initialized and SCAN_GDS is called. SCAN_GDS checks each field within the GDS related with the graph type. If the field contains graph data, the values within the elements of the field are passed to GRAFMAKER. If the field is empty, meaning the user has not yet specified the graph element, SCAN_GDS skips the field and checks the next field until all graph element fields related to the graph have been checked. This sequence is shown in Figure 51.

GRAFMAKER then renders the graph with the data contained in the "new" GRAFMAKER picture data structure. Since the GRAFMAKER picture data structure is initialized at every stage of the graph design, no attempt is being made to modify or alter the composition of an existing picture data structure.

Changes to graph element attribute values are only applied to the EZDRAW GDS. In this manner, the dynamic modification of graph elements is allowed.

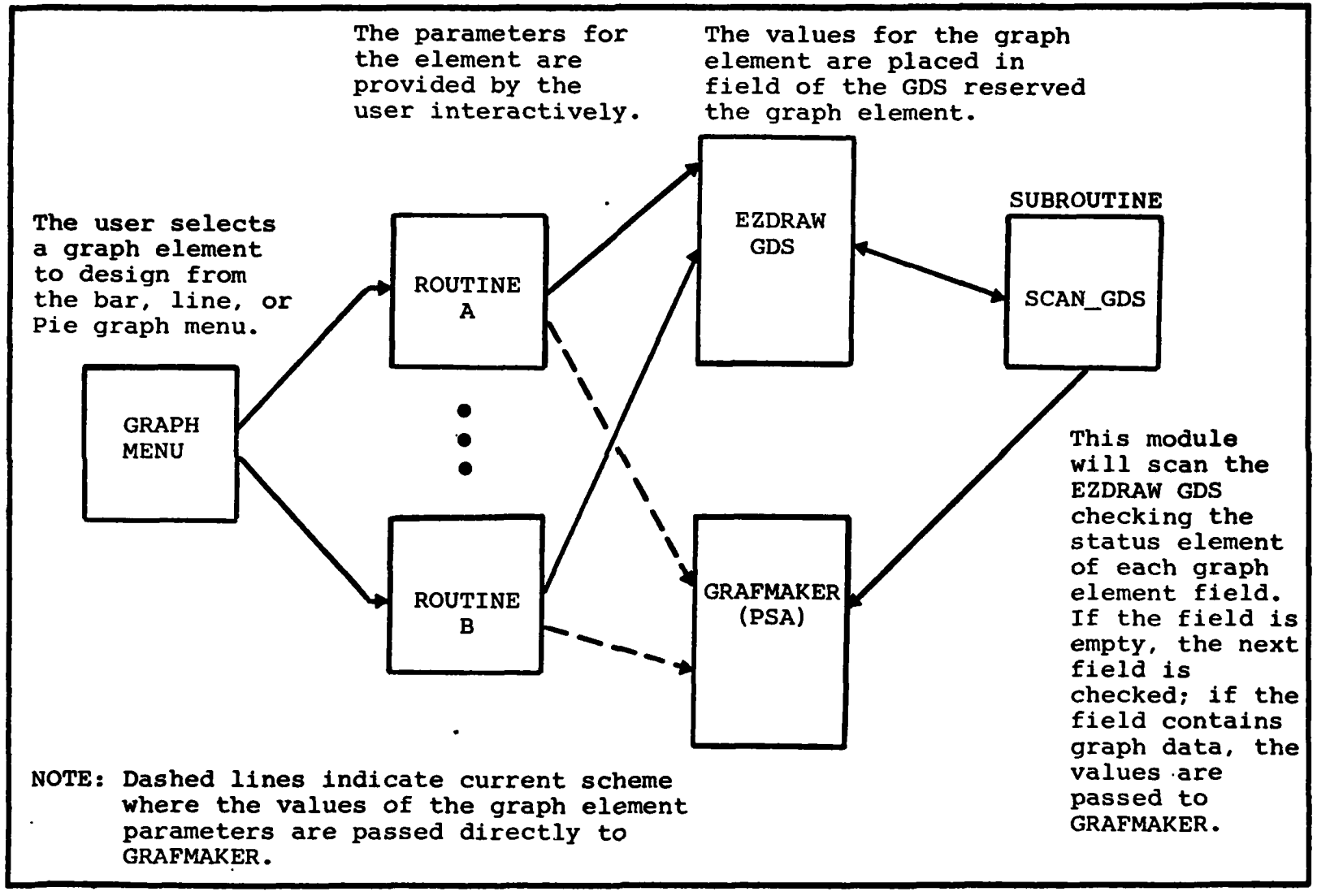


Figure 51. Implementing the GDS into EZDRAW

The EZDRAW Graph Data Structure

The GDS is envisioned to be a one-dimensional composite (integer and real) array. This will be accomplished by declaring the GDS in a common block as an integer array (IGDS) and a real array (RGDS) with equal dimensions. The two arrays will then be declared equivalent thus existing in the same memory location(s).

Figures 52a - 52c denote the structure of the proposed GDS. The 48 fields identified correspond to all possible graph elements supported by EZDRAW and a general purpose field which can be used with the recommended graph file directory.

The composition of each field is described further in Figures 53 - 60. In these figures, each element is defined along with its relative position within the GDS. An "R" notation within the element indicates the value is real. Otherwise, the element contains integer data.

In general, the first element in each field consists of a graph element status index. Initially set to "0", it is reset to "1" when the user selects the corresponding graph element from the graph menu. The purpose of this status element will be explained later. The remaining elements consist of the values for the parameters describing the graph element. The values are entered into the GDS in the following manner.

EZDRAW Graph Data Structure

FIELDS	
1	General Purpose
2	Horizontal Axis
3	Vertical Axis
4	Horizontal Axis Tick Marks
5	Vertical Axis Tick Marks
6	Curve #1
7	Curve #2
8	Curve #3
9	Bar #1
10	Bar #2
11	Bar #3
12	Horizontal Axis Label
13	Vertical Axis Label
14	Horizontal Axis Tick Mark Labels
15	Vertical Axis Tick Mark Labels
16	Curve/Bar #1 Label
17	Curve/Bar #2 Label
18	Curve/Bar #3 Label
19	Main Title
⋮	

Figure 52a. EZDRAW GDS

EZDRAW Graph Data Structure

FIELDS	
20	Sub-Title #1
21	Sub-Title #2
22	Note #1
23	Note #2
24	Note #3
25	Pie Attributes
26	Pie Segment #1
27	Pie Segment #2
28	Pie Segment #3
29	Pie Segment #4
30	Pie Segment #5
31	Pie Segment #6
32	Pie Segment #7
33	Pie Segment #8
34	Pie Segment #9
35	Pie Segment #10
36	Pie Segment Label #1
37	Pie Segment Label #2
38	Pie Segment Label #3
⋮	

Figure 52b. EZDRAW GDS

EZDRAW Graph Data Structure

FIELDS	
39	Pie Segment Label #4
40	Pie Segment Label #5
41	Pie Segment Label #6
42	Pie Segment Label #7
43	Pie Segment Label #8
44	Pie Segment Label #9
45	Pie Segment Label #10
46	Pie Segment Label Option Index
47	Exploded Pie Segments
48	Data

Figure 52c. EZDRAW GDS

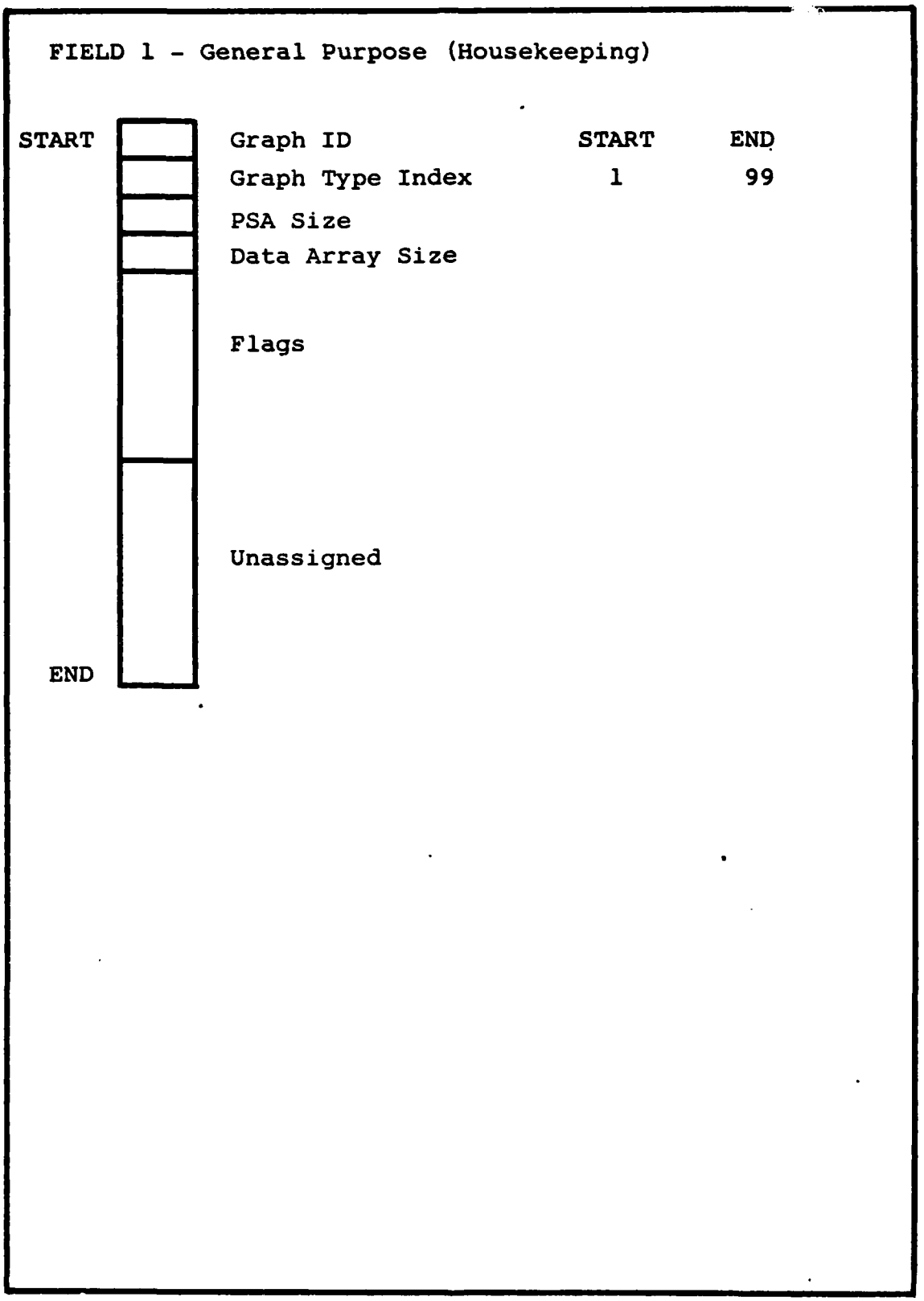


Figure 53. GDS Field 1

FIELDS 2/3 - Horizontal/Vertical Axis

			AXIS	START	END
START		Status			
		Axis #	Horizontal	100	107
		Line Index #	Vertical	108	115
		Axis Type			
		Line Color			
		Line Intensity			
		Line Style			
END		Line Width			

FIELDS 4/5 - Axes Tick Marks

			AXIS	START	END
START		Status			
		Tick Mark #	Horizontal	116	126
		Line Index #	Vertical	127	137
		Line Color			
		Line Intensity			
		Line Style			
		Line Width			
		Tick Mark Type			
	R	First Tick Mark			
	R	Last Tick Mark			
END	R	Tick Mark Increment			

Figure 54. GDS Fields 2,3,4,and 5

FIELDS 6,7, and 8 - Curve #1, #2, and #3

			CURVE	START	END
START		Status			
		Curve #	#1	138	145
		Line Index #	#2	146	153
		Line Color	#3	154	161
		Line Intensity			
		Line Style			
		Line Width			
END		Data Marker			

FIELDS 9,10, and 11 - BAR #1, #2, and #3

			BAR	START	END
START		Status			
		Bar #	#1	162	172
		Line Index #	#2	173	183
		Line Color	#3	184	194
		Line Intensity			
		Line Style			
		Line Width			
		Interior Color			
		Interior Pattern			
	R	Bar Width			
END	R	Bar Displacement			

Figure 55. GDS Fields 6,7,...,11

FIELDS 12/13 - Horizontal/Vertical Axis Labels
16,17,and18 - Curve/Bar #1, #2, and #3 Labels
19,20,and21 - Main Title, Sub-Title #1 and #2
22,23,and24 - Note #1, #2, and #3

START		Status
		Text Index #
	(84)	Converted Text String
		Font
		Color
		H-Justification
		V-Justification
		Path
		Quality
	R	Gap
	R	Ratio
	R	Rotation
	R	Size
	R	Width
	R	Height
	R	X-Position
END	R	Y-Position

TEXT STRING	START	END
H-Axis Label	195	294
V-Axis Label	295	394
Curve/Bar #1	597	696
Curve/Bar #2	697	796
Curve/Bar #3	797	896
Main-Title	897	996
Sub-Title #1	997	1096
Sub-Title #2	1097	1196
Note #1	1197	1296
Note #2	1297	1396
Note #3	1397	1496

Figure 56. GDS Fields 12,13, and 16,17,...,24

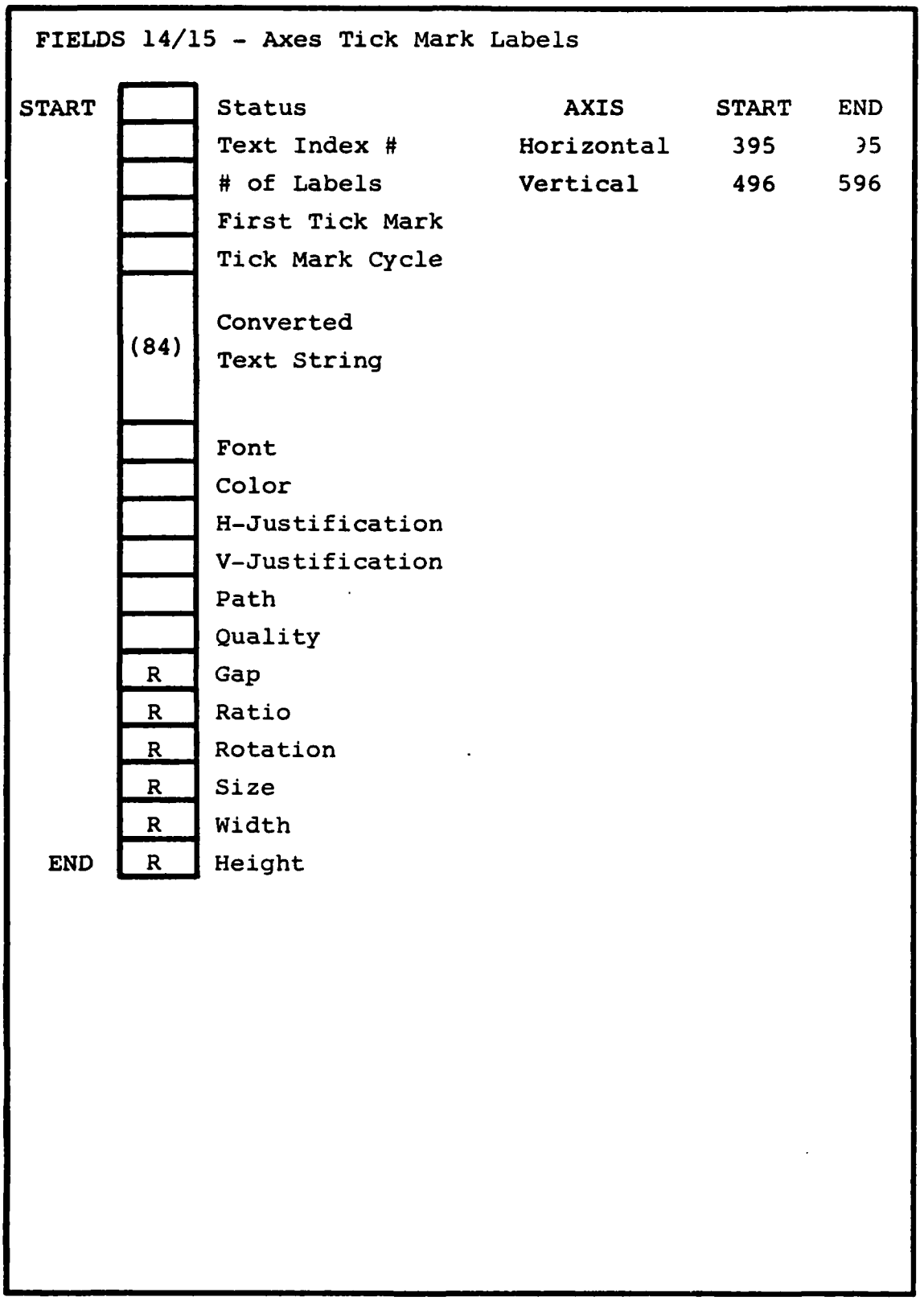


Figure 57. GDS Fields 14 and 15

FIELDS 26,27,...,35 - Segment #1, #2, ..., #10

			SEGMENT	START	END
START		Status			
		Segment #	1	1502	1510
		Line Index #	2	1511	1519
		Line Color	3	1520	1528
		Line Intensity	4	1529	1537
		Line Style	5	1538	1546
		Line Width	6	1547	1555
		Interior Color	7	1556	1564
END		Interior Pattern	8	1565	1573
			9	1574	1582
			10	1583	1591

FIELDS 36,37,...,45 - Segment #1, #2, ..., #10 Label

			SEGMENT	START	END
START		Status			
		Segment #	1	1592	1787
		Text Index #	2	1788	1883
			3	1884	1979
	(84)	Converted Text String	4	1980	2075
			5	2076	2171
			6	2172	2267
		Font	7	2268	2363
		Color	8	2364	2459
		Path	9	2460	2555
		Quality	10	2556	2651
	R	Gap			
	R	Ratio			
	R	Size			
	R	Width			
END	R	Height			

Figure 58. GDS Fields 26,27,...,45

FIELD 25 - Pie Attributes

START		Status	START	END
	R	X-Position	1496	1501
	R	Y-Position		
	R	Radius		
	R	Angle		
END		Direction		

FIELD 46 - Pie Segment Label Option

2652		Index	1 - Label only 2 - % only 3 - Value only 4 - Label & % 5 - Label & Value 6 - % & Value 7 - All

FIELD 47 - Exploded Pie Segment

START		Status	START	END
		Pie Segment 1	2653	2663
		Pie Segment 2		
		Pie Segment 3		
		Pie Segment 4		
		Pie Segment 5		
		Pie Segment 6		
		Pie Segment 7		
		Pie Segment 8		
		Pie Segment 9		
END		Pie Segment 10		

Figure 59. GDS Fields 25, 46, and 47

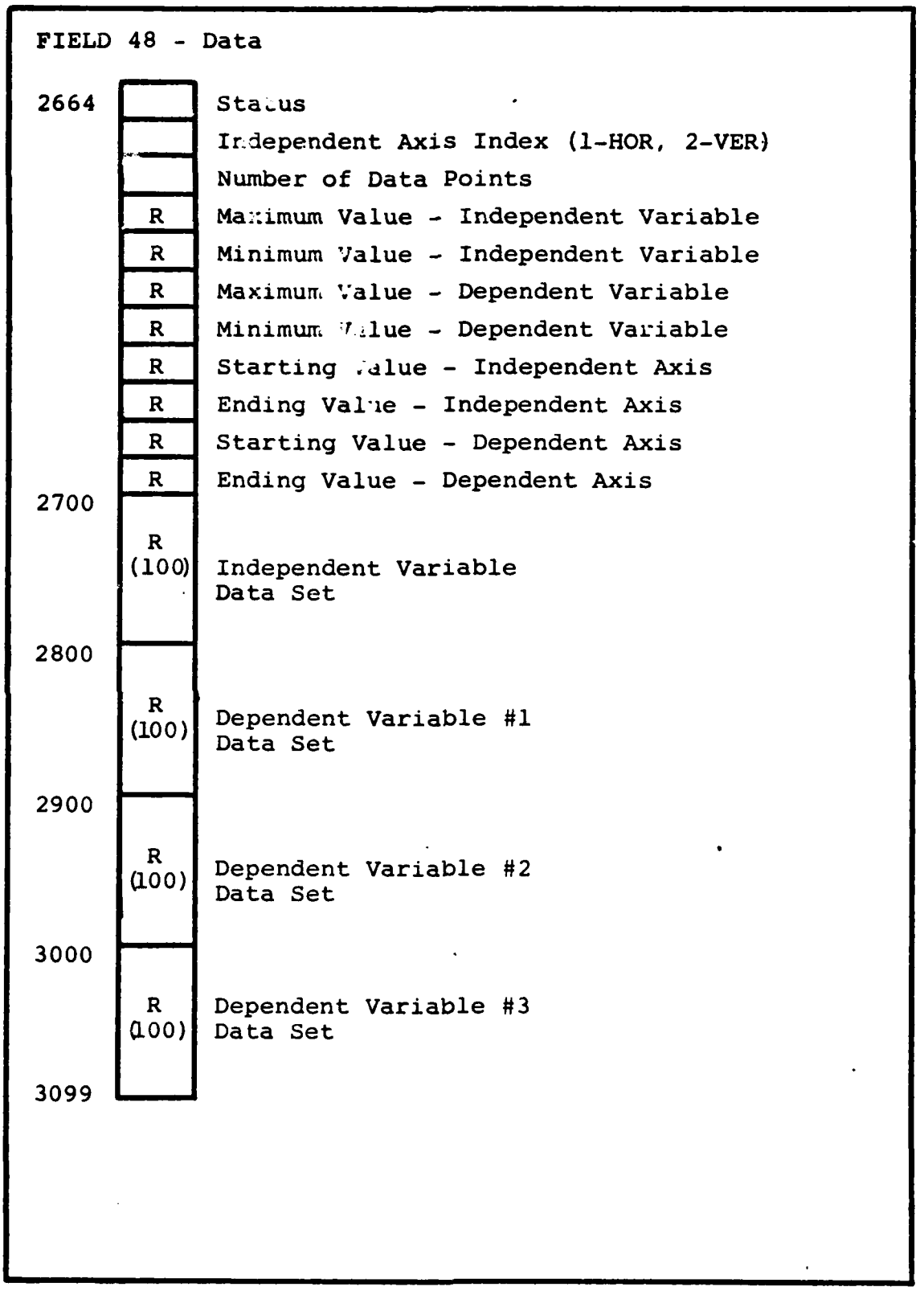


Figure 60. GDS Field 48

Suppose the user decided to design the horizontal axis for a line graph. EZDRAW calls the horizontal axis specifying routine (HORAX) and prompts the user for axis parameter values. As the user provides the values (or accepts current default values) EZDRAW places the values into the appropriate element within the horizontal axis field of the GDS (positions 100 - 107 of field 2).

EXAMPLE:

```
IGDS(100) = 1       /*field now contains data*/
IGDS(101) = 1       /*h-axis id #*/
IGDS(102) = 1       /*h-axis index #*/
IGDS(103) = AXTYPE /*type axis*/
IGDS(104) = COLOR  /*line color*/
IGDS(105) = INTENS /*line intensity*/
IGDS(106) = LNSTYL /*line style*/
IGDS(107) = LNWIDE /*line width*/
```

After these values have been placed in the field, SCAN_GDS is called. This component is now explained.

Subroutine SCAN_GDS

The purpose of this subroutine is to determine which parameters to pass to GRAFMAKER. Since the GDS supports each graph type, SCAN_GDS will first check the value of the graph type index (the second element in field 1). If the index is "1", the subroutine will only scan those fields in the GDS which contain data pertinent to line graphs. Bar graph related fields are scanned if the index is "2" while the fields related to pie graphs are scanned if the index is "3".

After EZDRAW initializes the GRAFMAKER picture data structure, SCAN_GDS checks the status index of the first line graph (per the example) field. If the status index is "0", then SCAN_GDS proceeds to the second field. Continuing with the example, suppose the second field to be checked is the horizontal axis field. SCAN_GDS proceeds as follows.

EXAMPLE:

```
/*determine if field is empty*/

     IF (field status is "1") THEN
        pass GDS field element values to GRAFMAKER
     END IF

     check next line graph field
```

The values for the horizontal axis attributes are passed to GRAFMAKER with these calls to GRAFMAKER subroutines.

```
/*define the line attribute represented by color,*/
/*  intensity, style, and width*/

      CALL JXLINE (PSA,PSIZE,IGDS(102),IGDS(104),
     +             IGDS(105),IGDS(106),IGDS(107))

/*provide line index number and axis attribute*/

      CALL JAXATR (PSA,PSIZE,1,IGDS(101),IGDS(102),
     +             0.0,IGDS(103))
```

In these calls, PSA and PSIZE are EZDRAW variables representing the GRAFMAKER picture data area and its dimension. In the call to JAXATR, the third argument is the Graph ID Number assigned to the one graph defined in the picture data area. The "0.0" corresponds to a scaling factor which has not yet been implemented in GRAFMAKER.

If one of the arguments in the field was a real value, then it would be passed as RGDS(n) and the appropriate real value would be passed to GRAFMAKER.

SCAN_GDS would then check the third field and so on until all fields containing data for line graphs were checked.

Implementing Considerations

All the graph element specifying routines in EZDRAW would have to be changed to place the values into the GDS rather than passing them directly to GRAFMAKER. While this is a non-trivial effort, it has been simplified somewhat since all these routines have been grouped together as a separate file. Appendix F (GRAFMAKER Subroutines Called by EZDRAW) and Appendix D (EZDRAW Subroutines) provide subroutine cross reference maps as additional aids. All GRAFMAKER subroutines are prefixed with a "J" sentinel further identifying the routines to be replaced.

The sequence in which the graph elements are specified must be consistent with the graph drawing sequence provided in the EZDRAW User's Manual. For example, if the user specifies the labels for tick marks before the tick marks have been specified, then these parameters are passed to GRAFMAKER by the GDS, and abnormal termination will result since GRAFMAKER requires specification of the tick marks to the tick mark labels.

Again, EZDRAW provides a built-in feature that prohibits

this from occurring. A system of element status flags are checked to determine if the designer attempts to draw graphs out of sequence.

The sequence of fields to be checked by graph type must also take into consideration this graph drawing sequence. SCAN_GDS must check the fields in accordance to the prescribed sequence. If the first required element has not been specified, fields of elements dependent upon the first element will not be checked.

Finally, EZDRAW must be analyzed to determine where the EZDRAW GDS must replace the GRAFMAKER picture data structure. Besides the graph designing operations, the GDS will be required in the graph saving and graph retrieval operations. Currently, EZDRAW saves the GRAFMAKER picture data structure. Under the proposed scheme, EZDRAW must save the GDS since it now contains all the pictorial data.

This proposed modification feature does have two drawbacks. First, the advantages gained in using the built-in GRAFMAKER picture data structure and manager are diminished. Secondly, the initialization of the GRAFMAKER picture data structure each time the graph is drawn will slow down the graph design process. However, the ability to modify the graph elements in this manner is too desireable and outweighs these considerations.

Appendix H

EZDRAW User's Manual

EZDRAW

An Interactive

Computer Graphics Program

To Design

Bar, Line, or Pie Graphs

PREFACE

This manual is based on Version 1.3 of EZDRAW. It is designed to provide the EZDRAW User a reference to the EZDRAW prompts and specific user requirements.

Much of the material in this manual comes from reference materials of the supporting graphics software (GRAFMAKER and DI-3000) from Precision Visuals. These packages are explained in this manual.

This manual reflects only what is currently available with the EZDRAW system. It does not address future enhancements, planned or on going.

CONTENTS

INTRODUCTION

Welcome to EZDRAW. This graphics package gives you the capability to custom design line, bar, or pie graphs in a totally interactive environment.

Some of EZDRAW's features include:

- Interactive/External Data File Graph Data Entry
- Two Dimensional Multi-variable Line or Bar Graphs
- Variable Character and Multi-Font Text
- "Exploded" Pie Segments
- Color Options for all Graph Elements
- Interior Shading/Pattern for Bars and Pie Segments
- Capability to "Save" Graphs for Future Use

HOW TO USE THIS MANUAL

A reading of this manual before you start using EZDRAW is not necessary. You will find that this manual is not technically oriented. Additionally the prompts, menu entries, and messages encountered within EZDRAW are straight forward, descriptive, and informative. This is characteristic of the entire EZDRAW system since EZDRAW was designed for the novice user (i.e. a person with little or no knowledge of computers, computer programming, computer graphics, or graph design).

The only assumptions made in EZDRAW are that the:

- User is familiar with a typewriter keyboard,
- Data for the graph is known, and
- User has a general idea or picture of what the desired graph should look like.

This manual is organized in much the same way as EZDRAW itself. A SET UP section is included to allow the user to use EZDRAW in one of three modes. Each major graph operation is explained in its own section in the manual. Each graph element that the user can design is also explained. The values (attributes) that can be assigned to particular graph elements are contained in a section. Finally, there is a section that provides recommendations on how to best use EZDRAW.

WHAT IS EZDRAW?

EZDRAW is an interactive computer graphics program aimed at allowing a novice user to design bar, line, or pie graphs. EZDRAW is menu and prompt driven. Therefore, only those options displayed or prompted can be entered.

EZDRAW was designed around the GRAFMAKER and DI-3000 graphics software packages written by Precision Visuals. The requirements imposed on the user are largely driven by the specifications of these packages.

GRAPHICS SOFTWARE SUPPORTING EZDRAW

EZDRAW is based on the following graphics software packages:

1. GRAFMAKER - a set of FORTRAN subroutines designed for use in preparing line, bar, or pie graphs. As such, EZDRAW is a GRAFMAKER application program which calls GRAFMAKER subroutines to specify the appearance of a graph to be output to one or more display devices.

2. DI-3000 - an integrated system of graphics software tools implemented in 1966 ANSI FORTRAN as a library of FORTRAN callable subroutines.

GRAFMAKER and EZDRAW call DI-3000 subroutines to generate graphics images on one or more display devices. In other words, GRAFMAKER allows the user to specify the values for elements of the graph while DI-3000 provides the actual graph drawing support.

EZDRAW REFERENCES

Within the EZDRAW HELP Mode and this manual, the user will frequently be referred to specific manuals. These reference prompts are presented as:

REF G:5-7, A-5 or
REF D:A-36, D-1.

In the first reference, the "G" stands for the GRAFMAKER User's Manual, pages 5 thru 7 and page 5 of Appendix A.

Similarly in the second reference, page 36 of Appendix A and page 1 of Appendix D in the DI-3000 User's Manual are depicted.

FATAL ERRORS

Due to the short period of time available to implement EZDRAW with GRAFMAKER (i.e. GRAFMAKER was installed in the fourth week of the research quarter) many aspects of GRAFMAKER's data handling routines are untested. Other untested conditions may arise from unexpected user input due to the interactive environment of EZDRAW. These unforeseen and untested conditions may cause EZDRAW and the supporting software to terminate abnormally causing the loss of all processing unless the graphical data has been saved as a graph file.

This should be an infrequent occurrence since EZDRAW has built-in error checking in all user input functions and consistently checks whether data being used for graphs is valid and appropriate.

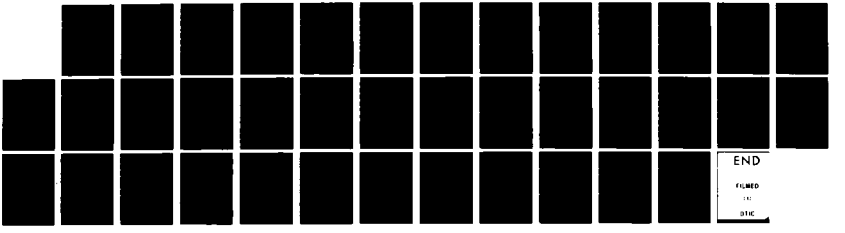
In the event that a "FATAL ERROR" occurs, the system will provide an error message to the user. This error message will consist of an error number and a brief summary of what may have caused the error. The error numbers and descriptions are found in the GRAFMAKER and DI-3000 User's Manuals (REF G:C-1, D:C-1).

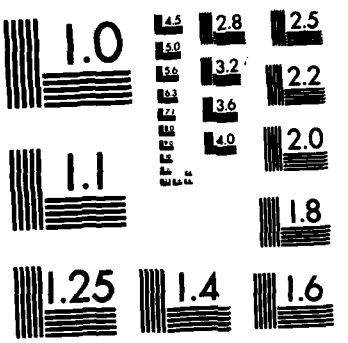
NON-FATAL ERRORS

In those situations where EZDRAW has correctly foreseen possible error conditions based on interactive user input, EZDRAW has its own set of error messages. Unlike the GRAFMAKER and DI-3000 errors, these are "NON-FATAL" and allow the user to recover from the processing error. Processing will continue at a logical point (i.e. re-entry of the required user input to a prompt or a menu).

AD-A124 901 EZDRAW AN INTERACTIVE COMPUTER GRAPHICS PROGRAM TO DESIGN BAR LINE OR PIE GRAPHS(U) AIR FORCE INST OF TECH WRIGHT-PATTERSON AFB OH SCHOOL OF ENGI.. R VEA DEC 82 3/3

UNCLASSIFIED AFIT/GE/MA/82D-1 . F/G 9/2 NL





EZDRAW is a portable (it can be used on several graphics display devices) graphics program that can be used on any of the following display devices:

TEKTRONIX 4014/4016
RAMTEK 6212
DEC VT100 (with graphics emulation)

Depending on the device you are using, follow the appropriate set up procedures. It is assumed that EZDRAW is currently available with appropriate access privileges invoked. It is also assumed that the specific values for EZDRAW parameters (the size of data arrays, the size of the graph picture, etc.) have been entered and that the subsequent compilation has been successful (i.e. EZDRAW is ready to "run").

You should insure that any supporting graphics copier peripheral to the display device is operational if copies of the graphics output is desired. For the RAMTEK 6212, this graphics copier is the Model 4007 Matrix Color Graphics Camera. The TEKTRONIX 4631 Hard Copy Unit is the graphics copier for the TEKTRONIX 4014/4016. Graphics output can not be copied from the DEC VT100. Refer to the respective graphics copier manual for their operation.

SETTING UP THE DEC VT100 (with Graphics Emulation)

1. Power up the VT100 per device instruction manual.

2. LOG ON.

3. Link EZDRAW to the appropriate device driver library by entering the following after the "$" prompt followed by a carriage return "<RTN>".

   $LINK EZDRAW,DILIB/LIB,T4014/LIB,DILIB/LIB<RTN>

4. Enter the VT100 graphics emulation mode.

   $<ESC> <SHIFT> $
   <ESC> <SHIFT> *
   <ESC> <SHIFT> $
   <SET UP> 9 <SET UP>

5. Commence program execution.

   RUN EZDRAW <RTN>

SETTING UP THE TEKTRONIX 4014/4016

1. Power up the TEKTRONIX 4014/4016 per the device instruction manual.

2. Insure the device is initialized.

   <RESET PAGE>
   Pause and let the screen "energize".
   <RESET PAGE>

3. LOG ON.

4. Link EZDRAW to the appropriate device driver library.

   $LINK EZDRAW,DILIB/LIB,T4014/LIB,DILIB/LIB<RTN>

5. Commence program execution.

   RUN EZDRAW <RTN>

SETTING UP THE RAMTEK 6212

1. Power up the RAMTEK 6212 per the device instruction manual.

2. Insure the device is initialized.

   $RAMTEK <RTN>

3. LOG ON.

4. Link EZDRAW to the appropriate device driver library.

   $LINK EZDRAW,DILIB/LIB,RAMTK/LIB,DILIB/LIB<RTN>

5. Commence program execution.

   RUN EZDRAW <RTN>

The EZDRAW Command Level is the highest level in the EZDRAW system. It is from this level that all the EZDRAW options and graph operations are accessed. This level is characterized by the EZDRAW Command Menu shown below:

```
                EZDRAW Command Menu

Select an Option:

   0 - HELP Mode
   1 - DATA Entry
   2 - DESIGN New Graph
   3 - MODIFY a Graph
   4 - VIEW a Graph
   5 - SAVE Current Graph
  99 - EXIT EZDRAW

Enter 0 ... 5 or 99 then <RTN>
>
```

The menu entries are self explanatory. Entry of a menu option will lead the user to either sub-menus or specific prompts. Sub-menus are designed similar to the Command Menu. For example, the first option in all menus is always "0" the HELP Mode while the last option is always "99" the menu exit. Successive entries of "99" from lower level menus will always return the user to the Command Menu. Entering "99" from the Command Menu will result in a prompt asking the user if the current graph is to be saved since all graph data is lost once EZDRAW terminates.

All of the options available from the Command Menu are explained in respective sections of this manual.

HELP MODE

WHAT IS THE EZDRAW HELP MODE?

The HELP mode is a brief summary of the major elements of EZDRAW. It is not an EZDRAW tutorial. The purpose of the HELP Mode is to provide the EZDRAW user with information that may be required during an EZDRAW session.

The HELP Mode provides capsulized information and when appropriate, refers the user to a reference manual. These reference notations are explained in the General Information section of this manual.

Information in the HELP Mode is presented one "screen page" at a time. By entering a carriage return (noted as PRESS <RTN>), the next "screen page" of information is presented. When the "last page" has been displayed, the user is returned to a HELP Mode sub-menu. Sub-menus are successively exited by selecting the "99" option.

WHEN IS THE HELP MODE AVAILABLE?

The EZDRAW HELP Mode can be accessed by the user at most levels of EZDRAW. Since EZDRAW is menu and prompt driven, the HELP Mode is offered to the user:

- in EZDRAW menus (except the HELP Menu)
- as a specific prompt prior to entry into lengthy/ detailed routines in which a menu is not present.

At all times, entry into the HELP Mode is left as a user option.

To exit from the HELP Mode, the user must return to the main HELP Mode Menu. Entering the "99" option will either return the user to another EZDRAW Menu or continue processing at the point where the HELP Mode prompt was displayed.

GENERAL INFORMATION ABOUT DATA ENTRY

Graphs, being mathematical functions, require data. EZDRAW allows four data sets to be used: one data set for one independent variable and a data set for up to three different dependent variables.

Data may be input interactively from the terminal or from an external data file. Interactive input of data is recommended for small data sets (typically for bar and pie graphs). Large data sets should be input from an external data file (usually for line graphs).

Data values may be real or integer numbers. EZDRAW converts all data values to real numbers as GRAFMAKER treats all data in the graph "data space" as real numbers.

As data is entered, EZDRAW determines and saves the minimum and maximum values for both the independent and dependent variables. These are presented to the user when the data is "linked" to the graph being designed (data can be used for more than one graph) for axis and tick mark definition.

EXTERNAL DATA FILES AND EZDRAW DATA ARRAYS

Data files used by EZDRAW must be FORTRAN 77 sequential, formatted files. Each file is accessed by EZDRAW after the user has input the name of the data file. The first element of the file must be an index indicating the number of data values contained in the file or to be read from the file. The remaining elements of the file must be the data set for a single variable.

Each data value in the file must be on a separate line. If an "end-of-file" condition is reached before EZDRAW reads the entire data set, the file is "closed" and EZDRAW informs the user that an error occurred during the reading of the data file. In this manner, file integrity is maintained and subsequent readings of the data file begins at the start of the file (the file is "rewound").

The size of EZDRAW data arrays is defined by the integer variable parameter "size". It has been set as a system standard to allow the user to input a maximum of 100 data values per graph variable. If the number of data values entered into the arrays are less than the size of the array, only the number of data values entered are used for the graph.

LINKING DATA TO LINE/BAR GRAPHS

When designing line or bar graphs, the first activity should be to "link" (associate) the EZDRAW data arrays to the graph.

EZDRAW links all data arrays when this option is chosen provided the data sets defined for the independent and dependent variables are valid. If all elements of the independent variable data array are zero, then EZDRAW will not link any of the data arrays to the graph. If the independent variable data set is valid, then EZDRAW determines if any of the data sets for the dependent variable is valid. If all elements of the dependent variable data arrays are zero, then EZDRAW will not link the data arrays to the graph.

Since all references to axes, tick marks, bars and curves are related to data, if the data is null for either variable, then these graph elements can not be designed.

If the data sets for both variables are valid, then the user must specify the start and end values for the independent axis. EZDRAW provides the maximum/minimum values for the independent variable previously input.

a. For line graphs, the axis start/end values should equal the independent variable minimum/maximum values or be "rounded" values to give more appropriate tick mark values.

b. For bar graphs, the axis start value should be less than the independent variable minimum value and the axis end value should be greater than the independent variable maximum value. Since the bars are centered about their independent axis values, this allows the bars at the ends of the axis to be complete. Otherwise, these bars will be truncated at the end of the graph data space. Figure 62 depicts this notion.

Then EZDRAW prompts the user for the start and end values for the dependent axis. EZDRAW provides the user with the current minimum and maximum values for the dependent variables. For line or bar graphs, the start and end values for the dependent axis may be equal. The only consideration here will be the values of the tick marks the user wants to display on the dependent axis.

If the end value equals the start value for either axis, then EZDRAW allows GRAFMAKER to compute the axis start and end values (REF G:A-48, A-90).

LINKING DATA TO PIE GRAPHS

When designing pie graphs, the first activity should be to link the EZDRAW independent variable data array to the graph. EZDRAW only links the independent data array for pie graphs. The values in the array are the values for segments of the pie.

EZDRAW allows a maximum of ten pie segments to be drawn. Therefore, if the user inputs more than ten data values into the independent data array, EZDRAW will not link the data to the graph.

Additionally, if the independent data array contains values less than or equal to zero, EZDRAW will not link the data to the graph. Pie segments can not be defined with zero or negative values.

GRAFMAKER computes the angle in degrees of each pie segment based on the sum of all data values and the value of each element in the independent data array. The segment percentage is also computed from these values.

THE DATA ENTRY PROCEDURE

To actually enter data, the user must be in the EZDRAW Command Menu. The "1" option should be selected which will then place the user in the Data Entry Menu. Since EZDRAW allows four different data sets to be entered, the user must indicate the data set to be entered.

Regardless of which data set is being entered, the user then indicates if the data will be input interactively from the terminal or if an external data file contains the data. The user next indicates if the data will be real or integer.

If the data will be input interactively, the user must then indicate how many data values will be input. EZDRAW informs the user of the current maximum value that can be entered (i.e. the size of the data arrays). EZDRAW then prompts the user for each value until all the data points have been entered. The user is then returned to the Data Entry Menu.

If data is read from an external file, the user provides the name of the data file. EZDRAW then accesses the file and reads the number of data values in the file to determine if it does not exceed the size of the data arrays. The data is then read and the user is returned to the Data Entry Menu.

Once all data sets have been entered, the user should exit the Data Entry Menu to the EZDRAW Command Menu.

DESIGNING GRAPHS

GENERAL INFORMATION ABOUT DESIGNING NEW GRAPHS

EZDRAW allows a user to design line, bar, and pie graphs. Each of these graph types are explained in their respective sections.

When designing new graphs, the user should have a good idea of what the final graph will look like. A rough sketch of the proposed graph would be useful during the design stage.

Each graph consists of different elements. It is up to the user to indicate the desired appearance of each element. EZDRAW allows the user to select from a graph menu which element of that particular graph will be designed. The process of indicating the appearance of the element will be referred to as "specifying". A user will specify the values of the attributes of the element through the EZDRAW menus and/or prompts. The graph elements and their associated attributes are described in this manual.

When the attributes for the elements of the graphs are being specified, the user may want to know about possible entries before making the entry. The EZDRAW HELP Mode provides information about an element and its attributes. The HELP Mode is generally not available once the user actually enters the routines requiring user input. This is when having the EZDRAW User's Manual is useful.

GRAPH TYPES

Before selecting the type of graph to support the data to be represented, it is important to note that there are four basic types of relationships or patterns from which to choose in graph design. They are time series, parts of the whole, comparing several items, and relationships between two or more variables.

LINE GRAPHS. Time series, or the change over time of one or more dependent variables, is best displayed using the line graph. The line graph is also useful when explaining the relationship between an independent variable and one or more dependent variables.

With line graphs, multiple curves are usually plotted. Hence, the designer should consider how each curve will be compared with the other curves.

Curves should always be thicker than the axis. This makes the graph easier to read and delineates what part of the graph is the curve.

The data points themselves should be highlighted by the use of data markers. Different markers should be used for each curve. The use of too many points can clutter the graph and imply a precision that does not exist.

BAR GRAPHS. Bar graphs are useful when comparing several items and when attempting to show the relationships between two variables. A bar graph may be more effective than a line graph if there are few time periods. They are especially effective in showing large changes from one period to the next.

Judicious use of color, spacing, and patterns enhance the effect of bar graphs. The bars of the graph should always be wider than the space between the bars.

PIE GRAPHS. Pie graphs are generally used when noting parts of a whole or percentages.

Exploded pie graphs are useful when the user wants to focus the eye on one or more segments of the graph. The proper use of color and interior patterns of the pie segments can also enhance the pie graph.

Pie graphs with too many segments are difficult to comprehend. Therefore, the number of segments in pie graphs should be limited to four or five. EZDRAW allows a maximum of ten segments to be drawn.

EXITING THE DESIGN MODE

When designing a new graph, EZDRAW initializes the area within EZDRAW which contains the data for the graph. This area is called the picture storage area (PSA). It is also initialized by graph type. In this manner, only the elements for one particular type of graph may be input to the PSA.

When the user exits the design mode (to save the graph for example), then the only operations that can be performed on the graph are: Modify, View, or Save. If the user attempts to enter the design mode again, any work done on a graph will be with a new graph since EZDRAW has reinitialized the PSA.

To work on the graph again, it must be modified. See the MODIFYING GRAPHS section.

If the graph is not saved prior to exiting EZDRAW, then the graphical data in the PSA is lost and can not be recovered. It is best to save the graph at some stage of design and then continue work on the graph in the modify mode.

THE DIFFERENCE BETWEEN DESIGNING AND MODIFYING

Designing and modifying graphs are similar in that they both allow the user to specify the same graph elements of each type graph. In fact, internal to EZDRAW, the same graph element specifying routines are used in either mode.

The difference between the two modes is:

DESIGN - the PSA is initialized and prepared for a particular type of graph. In this manner, a "new" PSA is being used for the graph design.

MODIFY - once the desired graph has been retrieved (see RECALLING GRAPHS FOR MODIFICATION) from the graph file, the PSA contains the data for the graph at the point at which it was saved. In this manner, an "old" PSA is being used for the subsequent graph design.

THE DESIGN MODE

The design mode is accessed by entering option 2 from the EZDRAW Command Menu. EZDRAW then determines if the user wishes to save the contents of the current graph. If the graph is to be saved, the SAVE operation commences (see SAVING GRAPHS). If the graph is not to be saved, then the DESIGN Menu is displayed.

From the DESIGN Menu, the user has the option of designing one of the three graph types. After the user has entered the type of graph that is going to be designed, either the Line Graph Menu (Figure 61), Bar Graph Menu (Figure 62), or the Pie Graph Menu (Figure 63) is displayed. The menu lists those graph elements that can be designed by the user.

Along with the menu, is a rectangular area which is called the "drawing area". Whenever the user has finished designing an element of the graph, the product of the design is displayed in this drawing area providing immediate feedback.

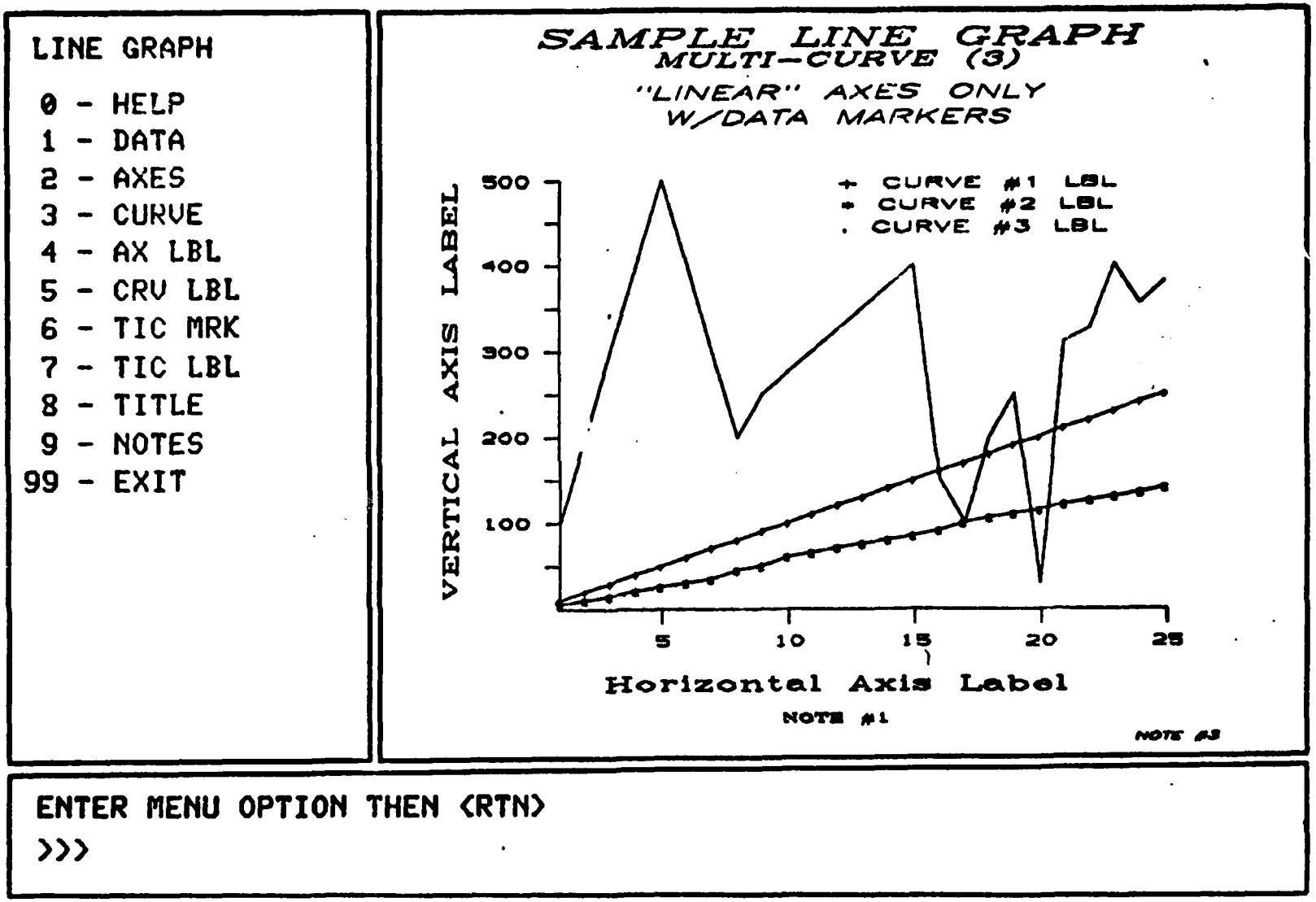


Figure 61. Sample EZDRAW Line Graph

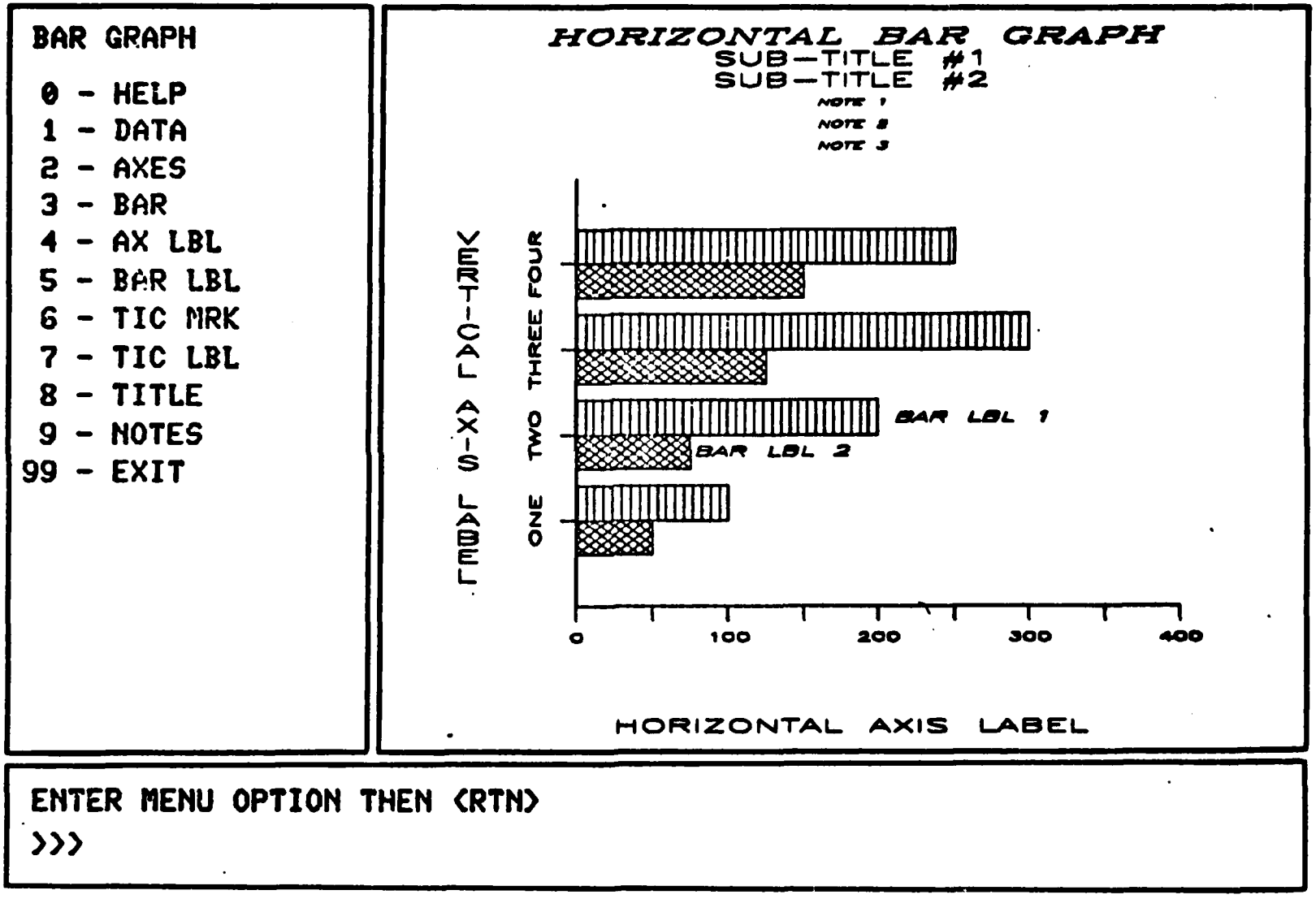


Figure 62. Sample EZDRAW Bar Graph

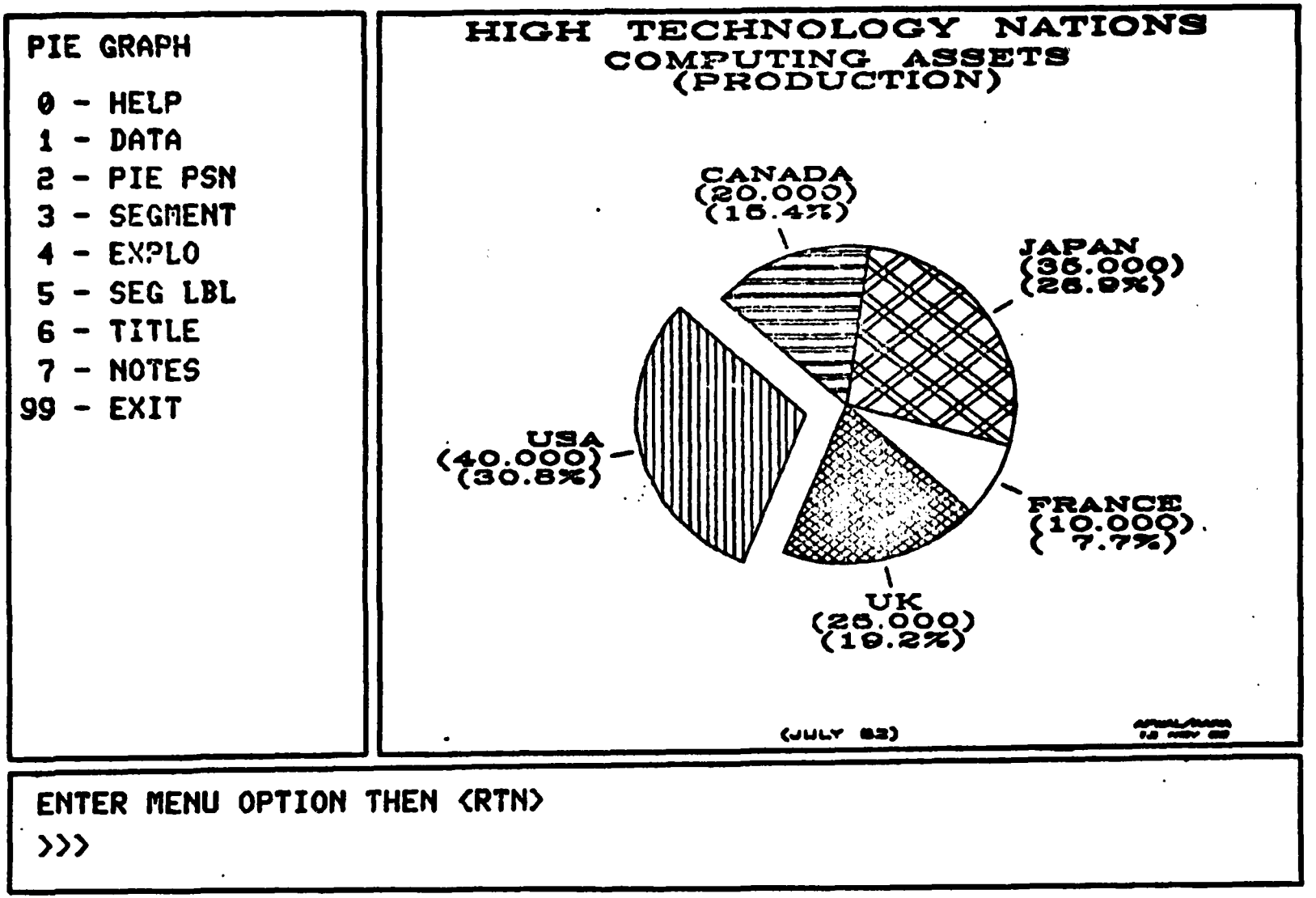


Figure 63. Sample EZDRAW Pie Graph

Below the menu and drawing areas is a prompt area which asks the user for the entry from the menu. Selecting an item from the menu then brings the user to the EZDRAW graph element specifying routines. They consist of sub-menus and prompts to which the user responds with the values to be assigned to the graph element. These graph elements and their associated attributes are explained in this manual.

Selecting option 99 from the graph menu will return the user to the EZDRAW Command Menu.

MODIFYING GRAPHS

GENERAL INFORMATION ABOUT MODIFYING GRAPHS

The Modification Mode is a means of changing a previously designed graph.

This version of EZDRAW does not allow the dynamic modification (changing the appearance of graph elements already specified) of graph elements that have already been specified in the initial Design Mode or previous modifications of the graph. Attempts to change a graph element that has already been specified will result in non-fatal EZDRAW error messages indicating the graph element has already been specified. The user will then be returned to the graph menu or a sub-menu for entry of a new option. This limit applies to whether the user is in the Modify or Design Mode.

Generally, the only modifications allowed are the addition of graph elements that have not been specified before. However, some changes are allowed. These allowable changes are limited to:

1. Repositioning of text strings that have already been specified. Attempts to reposition a text string that has not been specified will result in a non-fatal error message.

2. Redefinition of the following pie graph elements:

   Center of the pie (pie position within the graph),
   Radius of the pie,
   Direction of the pie, and
   Rotation of the pie.

These elements may be modified as often as desired by selecting option 2 from the Pie Graph Menu.

RETRIEVING PREVIOUSLY DESIGNED/SAVED GRAPHS

When the user enters the Modify Mode (selecting option 3 from the EZDRAW Command Menu), EZDRAW determines if the user wishes to modify the current graph or a graph that has been previously designed and saved.

If the user wishes to modify another graph, then EZDRAW determines if the user wishes to save the contents of the current graph in the PSA. If the user wishes to save the graph, EZDRAW automatically initiates the SAVE routine. See the SAVE section of this manual for a description of this operation.

After saving the graph or if the user decided not to save the graph, then the user must provide EZDRAW the file name and file record number to which the desired graph was saved. See RETRIEVING PREVIOUSLY DESIGNED/SAVED GRAPHS in the VIEWING GRAPHS section. After the desired graph has been retrieved, EZDRAW displays the graph and its associated graph menu.

Graphical processing may continue at this stage in the same manner as in the Design Mode.

VIEWING GRAPHS

GENERAL INFORMATION ABOUT VIEWING GRAPHS

EZDRAW allows the user to "view" (look at) the current graph (the graph being designed or modified) or another graph that was previously designed and saved in a graph file. When the desired graph is displayed, it is enlarged so that the display screen is maximized for size and resolution.

EZDRAW prompts are not visible on the enlarged display of the graph. In this manner, a copy of the graph may be made on the graphics copier peripheral to the graphics display device being used for graphics output.

RETRIEVING PREVIOUSLY DESIGNED/SAVED GRAPHS

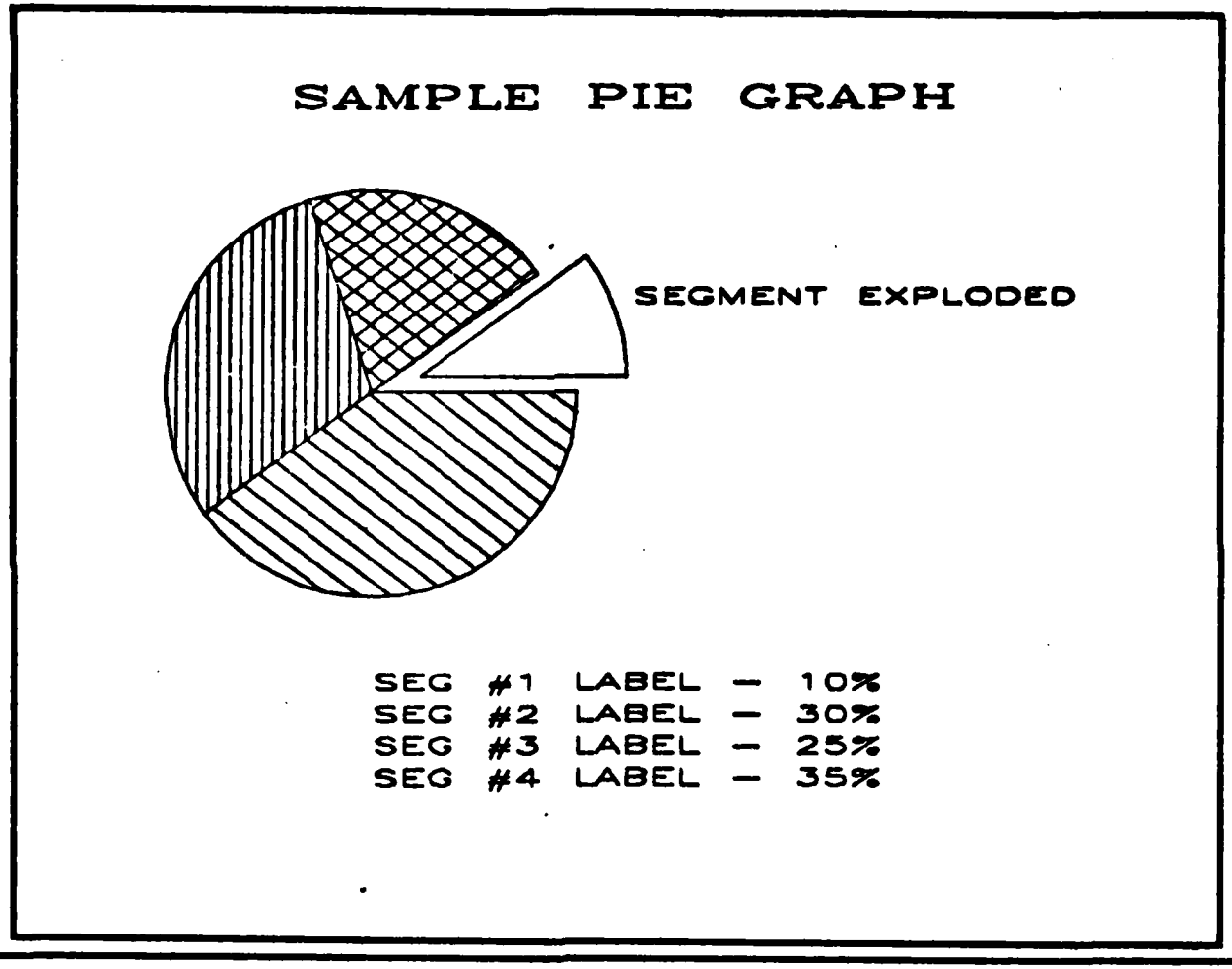
If the user wants to view a previously created and saved graph that is not the current graph, then the user must provide EZDRAW the exact name of the graph file in which the graph was saved. The file specified by the user is then accessed and opened by EZDRAW.

EZDRAW then prompts the user for the record number of the graph within the file.

If the record number entered is greater than the last graph record saved to the file, then EZDRAW will provide an error message indicating that it attempted to read a record that did not exist (i.e. the records in the file are 1,2,3...8 and the user directed EZDRAW to retrieve record 9 or greater; then the error message will be displayed).

If the record number entered is a number less than the last record number but was not a record number saved by EZDRAW (i.e. the records in a file are 1,2,3...6 and the next graph is saved as record 9 in the file; then the supporting system "creates" records 7 and 8) then EZDRAW will not retrieve the graph since the record did not contain valid graphical data.

When a valid record number is provided, the data in the record is transferred to EZDRAW. When this transfer of data occurs, this graph is now considered the current graph and is displayed to the user with a prompt to determine if the correct graph was retrieved. This is depicted in Figure 64.



IS THIS THE CORRECT GRAPH ?
ENTER Y OR N THEN <RTN> -

Figure 64. Retrieved Graph Displayed

a. If the correct graph was retrieved, then EZDRAW informs the user that the graph will be displayed by entering a carriage return. The user is also informed that when viewing or copying is complete, another carriage return will allow EZDRAW to continue by placing the user at the EZDRAW Command Menu.

b. If the correct graph was not retrieved, then EZDRAW continues to ask for the graph file name and file record number until the correct graph has been retrieved.

MAKING COPIES OF GRAPHICS OUTPUT

When the desired graph is displayed enlarged, hard copies of the graphics output may be made with the graphics copier attached to the graphics display device being used.

On the RAMTEK 6212, this is the Model 4007 Matrix Color Graphics Camera. On the TEKTRONIX 4014/4016, this is the TEKTRONIX 4631 Hard Copy Unit. The appropriate operating manuals for these units should be referenced.

On the VT100, copies of the graphics output can not be made.

SAVING GRAPHS

GENERAL INFORMATION ABOUT SAVING GRAPHS

EZDRAW allows the user to save line, bar, or pie graphs that have been designed for later viewing, copying, or modification.

Graphs are saved in files that are user specified. The actual graph is saved as a particular record within the file. Each file may contain up to 999 records. The record to which the graph is saved, is specified by the user.

The graphical data that is actually saved in the file is what has been specified and defined by the user during the graph element design stage.

Graphs may also be saved as a DI-3000 METAFILE. The DI-3000 METAFILE system is another means of saving graphs for later manipulation. See the METAFILE section of this manual for additional information.

GRAPH FILES

EZDRAW saves graphical data in files specified by the user. The actual data is located within EZDRAW in a variable called the picture storage area (PSA). The PSA is an integer array that can be given a new dimension prior to recompiling EZDRAW for use with graphs with very large data sets.

Files are created by EZDRAW using standard FORTRAN 77 file operations. The files created by EZDRAW are direct access and unformatted. The files consist of an index indicating the type of graph was saved, an array of status flags indicating which graph elements have been designed, and the picture storage area.

When saving graphs to a file, the user is required to provide the name of the file in which the graph data will be saved.

File names consists of a basic name (up to six characters), a period ".", and an extension (up to three characters). Examples of file names are:

BASE.EXT, GRAPHS.DAT, PIEGRF.ONE

The supporting computer system will append a version number to the file name (e.g. BARGRF.TWO;# where # is the version number) if a file name is used more than once.

FILE RECORD NUMBERS

Each graph that is saved to a graph file, must be saved as a particular record within the file. The record number is provided by the user and must be a positive integer number between 1 and 999. This implies that only 999 records can be saved in one file.

Refer to the CAUTIONS section for more information about file numbers.

NEW GRAPH FILES

After the user has specified the name of the file to which the graph will be saved, EZDRAW will ask the user if the file already exists. If not, the file is considered a "new" file and EZDRAW will create the file for the user.

EZDRAW then informs the user that the file has been created by reiterating the file name that the user has provided.

Subsequent references to this file to the EZDRAW "new file" prompt should be answered by specifying the file as "old". However, if the user indicates that the file is new, EZDRAW will create a newer version of the file while maintaining the "old" version.

DI-3000 METAFILES

After the user has saved the graph in a graph file, the user is given the option of saving the same graph in the DI-3000 METAFILE system.

If the user decides to save the graph as a METAFILE; then the graph is displayed on the screen. This is the means in which the graphical data is transferred to the METAFILE Generator. The names of graphs saved in this manner are METAFILE.MFL;# (where # indicates version number). The user is then returned to the EZDRAW Command Menu.

Graphs saved in the METAFILE can be accessed at a later date by using the DI-3000 METAFILE Translator. This is an

interactive program that post-processes DI-3000 METAFILEs with graphics output to selected devices. The translator may read up to 5 METAFILEs concurrently. They may be positioned, scaled, and superimposed on the selected graphics device (Ref D:D-1).

CAUTIONS/RECOMMENDATIONS WHEN SAVING GRAPHS

1. When designing/modifying graphs, the graph should be saved at various stages of its design. As a general rule, it is prudent to have a backup copy of the graph.

2. File names should be descriptive of the graph that is being saved.

3. Each graph should be saved in its own file. In this manner, file and graph management is simplified.

4. When saving graphs at their various stages of design, it is best to save the graph in consecutive records within the file. If you save the first iteration as record 1 and then save the next as record 3, FORTRAN creates a "dummy" record 2. As previously explained, "dummy record 2" does not contain any graphical data.

5. Be aware that if a graph is saved as a record that already exists in the file (i.e. saving a graph as record 5 while record 5 already exists in the file), the new data will "over-write" the old data of record 5.

6. Graphs should not be saved in existing files whose record length (the size of the PSA) is different from the record length of the records within the file.

7. EZDRAW error messages indicate to the user when a discrepancy occurs in the file operation. The user should be knowledgeable of the existing graph files and their contents. Until a more efficient file system can be devised, the user should therefore keep a listing of the graphs saved, the file to which it was saved, the record number of the particular graph, and the size of the PSA.

GRAPH ELEMENTS

All of the graphs allowed by EZDRAW are composed of several elements. An element is defined to be a part of a graph that has particular attributes associated with it that must be specified by a user option or system standard value.

The elements of the graphs can be grouped into three broad areas: lines (linear elements), text, or data.

Each of the graphs contain elements that are unique to that particular graph while others are common to all the graphs. The elements for each graph type are noted below and are displayed in the sample graphs of Figures 61, 62, and 63.

LINE GRAPH ELEMENTS

LINE ELEMENTS

1	Horizontal axis
1	Vertical axis
1	Group of major tick marks along the horizontal axis
1	Group of major tick marks along the vertical axis
1-3	Curves

TEXT ELEMENTS

1	Main title
2	Sub-titles
1	Horizontal axis label
1	Vertical axis label
1	Horizontal axis tick mark label group
1	Vertical axis tick mark label group
1	Label per curve on the graph
3	Additional notes

DATA ELEMENTS

A data marker for each data point on the graph. Each curve can have a unique marker for the set of data points along the curve.

BAR GRAPH ELEMENTS

LINE ELEMENTS

1	Horizontal axis
1	Vertical axis
1	Group of major tick marks on the horizontal axis
1	Group of major tick marks on the vertical axis
1-3	Border/edge of up to 3 different bar groups

TEXT ELEMENTS

1	Main title
2	Sub-titles
1	Horizontal axis label
1	Vertical axis label
1	Horizontal axis tick mark label group
1	Vertical axis tick mark label group
1	Label per bar group on the graph
3	Additional notes

DATA ELEMENTS

Width of the bar within each bar group.
Displacement of a bar from a tick mark along the independent axis.
Interior color/pattern of the bar.

PIE GRAPH ELEMENTS

LINE ELEMENTS

The border/edge of each segment of the pie graph.

TEXT ELEMENTS

1	Main title
2	Sub-titles
3	Additional notes
1	Label per pie segment
1	% value per pie segment (user option)
1	Pie segment magnitude value (user option)

DATA ELEMENTS

Interior color/pattern of pie segments
Exploded pie segments
Radius of the pie
Direction of the pie
Rotation of the pie
Position of the pie

GENERAL INFORMATION ABOUT ATTRIBUTES

The attributes of a graph element are those values that define the appearance of the element (i.e. text color, line style, etc.).

In a majority of cases, the value for the attribute is user specified. Values that are system standards are defined below.

An explanation of each attribute by graph element type is provided as well as the reference in the GRAFMAKER and/or DI-3000 User's Manual.

ATTRIBUTES FOR GRAPH LINE ELEMENTS

LINE INTENSITY. Line intensity defines the relative brightness of all line elements. The user may select the line intensity from a range of 0 to 32767. The EZDRAW default is 18383. (REF G:A-94, D:26)

LINE WIDTH. Line width is the relative thickness of line elements. The user may select the line width from a range of 0 to 32767. The EZDRAW default is 18383.

LINE STYLE. Line style refers to the appearance of the graph line element (solid, dashed, dotted, etc.).

The principal element that should consider this attribute are the curves of line graphs for differentiation. All other line elements should be solid.

The user may select line styles from a range of 0 to 32767 (0 is a solid line and the EZDRAW default). This attribute is highly device dependent. Line styles 0, 4, and multiples of 4 are solid lines. (REF G:A-94, D:26)

LINE COLOR. The user may designate the color of line elements to be red, green, yellow, blue, magenta, cyan, white, and a "normal" color based on the display device. The EZDRAW color default is "normal". (REF G:A-42, D:25).

Color should be consistent between these elements:

Horizontal axis, vertical axis, and tick marks,
Borders and interiors of bars,
Borders and interiors of pie segments,
Curves and their labels,
Bars and their labels, and
Pie segments and their labels.

ATTRIBUTES FOR GRAPH TEXT ELEMENTS

TEXT QUALITY. Quality refers to the precision or degree to which a text string, when output to a display device, adheres to the values of the text attributes. Different precisions/qualities do not follow all of the set text attributes. The user may select the text quality for text strings from those noted below.

1. String precision - lowest quality. This quality uses the hardware/software generator of a device. It adheres as closely as possible to text size, gap, and justification. All other attributes are ignored.

2. Character precision - medium quality. This quality uses the device level character generator. It adheres as closely as possible to the text size. The other attributes are adhered to exactly.

3. Stroke precision - high quality. Each character is stroke generated in software. It adheres exactly to all text attributes. Font is simple stick characters.

4. Graphics arts precision - highest quality. Each character is stroke generated in software. It adheres to all text attributes and has different character spacing and fonts.

The EZDRAW default is stroke precision. For titles, labels and notes, the quality on graphs should be graphics arts. (REF G:A-85, D:31-32).

TEXT COLOR. The user may designate the color of text strings to be red, green, yellow, blue, magenta, cyan, white, and "normal" for the device. The EZDRAW default is "normal". (REF G:A-42, D:25)

Colors should be consistent between these elements:

Related text groups (i.e. titles),
A curve and its label,
A bar and its label, and
A pie segment and its label.

TEXT FONT. Text font describes the print/character style to be used for characters of text strings. Only the graphics arts quality of text allows the user to select from 12 different fonts. Graphs should maintain integrity of font styles in all text elements. The EZDRAW default font is Block Simplex. (REF G:A-95, D:34-35.2)

TEXT PATH. Text path is an index representing one of four directions in which a user may indicate the text string to be written. These directions are right, down, left, and up. The normal direction and the EZDRAW default is right.

Although the axis label for the vertical axis can be specified to have a path of down, it is recommended that this label have a path right and be rotated 90.0 degrees. (REF G:A-84, D:33-34)

INTER-CHARACTER GAP. The character gap defines the spacing between adjacent characters in a text string in the direction of the character path. The inter-character gap is specified as a fraction of the average character size.

The range of user options is -1.0 to +1.0. A value of -1.0 implies complete character overlap and +1.0 implies double width between characters. The EZDRAW default is 0.0 (normal). (REF G:A-95, D:36)

TEXT JUSTIFICATION. Justification refers to a pair of indices with which the user defines how a text string will be aligned (horizontally and vertically) along a specific character path.

Text strings can be justified horizontally to the left, center, or right. The EZDRAW default for horizontal justification is center.

Text strings can be justified vertically from the bottom, center, or top. The EZDRAW default for vertical justification is center.

Justification is ignored for labels of pie graph segments. This is automatically done by GRAFMAKER based on the position of the segment.

The justification point for tick marks along the horizontal axis is one character height below the end point of the tick mark. Labels for tick marks along the horizontal axis should be justified either:

a. Horizontal - Center, Vertical - Top (no rotation)
b. Horizontal - Right, Vertical - Center (if rotated).

The justification point for tick marks along the vertical axis is one character width to the left of the end point of the tick mark. Labels for tick marks along the vertical axis should be justified either:

a. Horizontal - Right, Vertical - Center (if rotation is less than +/- 90.0 degrees)
b. Horizontal - Center, Vertical - Bottom (if rotation is 90.0 degrees).

(REF G:A-83, D:34-35)

WIDTH TO HEIGHT RATIO. Text ratio is a measure of the relative character width divided by the character height. EZDRAW has set this to be a constant 0.75. (REF G:A-95)

TEXT SIZE. Text size is the height of the characters of a text string. Text size is expressed in terms of the graphical space units in the "y-direction" of the graph. The user may select a size from a range of 0.0 to 760.0 units. The EZDRAW default is 18.0 graph units.

The following sizes are recommended for the noted text element:

Main title	35 - 45
Sub-titles	20 - 30
Axis labels	18 - 25
Tick mark labels	15 - 20
Curve/Bar/Segment labels	20 - 25
Annotations	18 - 25
Logos	10 - 15

Bear in mind that the size of the characters in the text string determines the maximum length of the text string (i.e. the maximum number of characters for the string). A larger text size will decrease the maximum length of allowable strings. Text strings that do not "fit" within the borders of the graph space are "clipped". (REF G:A-82)

TEXT ROTATION. Text strings are normally written horizontally to the right without being rotated. However, text strings may be rotated about their justification point at the option of the graph designer.

Rotation angles vary from 0.0 to +/- "N" degrees. The EZDRAW default rotation angle is 0.0 degrees. Angles greater than 0.0 are counter-clockwise while angles less than 0.0 are clockwise. Care should be exercised to insure that a text string is not rotated so much that it is oriented in an "upside-down" manner. (REF G:A-87)

ATTRIBUTES FOR GRAPH DATA ELEMENTS

Since any graph is basically a representation of data, the elements of the graph that are related to data require explanation. The data elements for each graph type have been explained previously. The attributes for those elements are now described.

DATA REPRESENTED ON LINE GRAPHS. EZDRAW allows up to three different dependent variables to be plotted against an independent variable per line graph.

The data points for these curves are input into the EZDRAW data arrays either interactively or from an external sequential data file. The data must then be linked to the graph from the Line Graph Menu (option 1).

When linking data, the user must specify the start and end points for both the independent and dependent axes. EZDRAW provides the user with the minimum and maximum values for these variables based on the data input. It is up to the user to:

1. Make the start and end values of the axes equal to the minimum and maximum values of the variables, or

2. Make the start and end values of the axes to be appropriate rounded values less than or greater than the maximum and minimum values of the respective variables, or

3. Let GRAFMAKER compute the start and end values for the axes by designating the start and end values to be equal.

There are several ways that the different curves can be differentiated from each other:

1. Use markers: ".", "+", "*", "0", "X", or none for each data point on the curve.

2. Use different colors for each curve.

3. Use different line styles for each curve.

The labels for the curves should be near the curve rather than in a legend. The label should contain the marker symbol of the curve (if markers were used) and/or be the same color.

DATA REPRESENTED ON BAR GRAPHS. EZDRAW allows the user to represent up to three different variables to be plotted against one independent variable. The data for the bars are input into the EZDRAW data arrays either interactively or from external sequential data files.

When the data is linked to the bar graph, the user must provide the start and end values for the dependent and independent axes. EZDRAW provides the user with the maximum and minimum values of the dependent and independent variables.

1. For the independent variable, it is critical that the user provide sufficient "data space" along the independent axis so that the end bars are not truncated. The user should make the axis start value less than the minimum value of the independent variable and make the axis end value greater than the maximum value of the independent variable.

2. For the dependent variable, the user should consider which data points will be represented by tick marks.

When data is first linked to the bar graph, the bars span the width of the data space since by default, the bar width is the width of the entire data space. The user must specify the width of the bar based on the independent variable data units.

1. If the independent variable ranges from 0.0 to 100.0 and is incremented in tick marks by 10.0, then the bar width should be less than or equal to 10.0.

2. The number of different bar groups will also have a bearing on the width of each bar. All bar groups should have equal widths for consistency in comparison.

By default, the bar is centered directly over a designated tick mark. If there are more than one bar groups, then all the bars will overlap, centered on the tick mark. Therefore, each bar group must be displaced from the tick marks. This displacement is specified by the user.

1. If the bars are vertical, a positive displacement will move the bars to the right of the tick mark while a negative displacement will move the bars to the left of the tick mark. A displacement of 0.0 will center the bar over the tick mark.

2. If the bars are horizontal, a positive displacement will move the bars above the tick mark while a negative displacement will move the bars below the tick mark. A displacement of 0.0 will center the bar to the right of the tick mark.

Displacement is expressed in data space units. It is best to displace the bars in terms of bar widths (assuming all bar groups are the same width).

1 bar group:	center the bar over the tick mark. The displacement is 0.0.
2 bar groups:	displace 1 bar group to the right/above the tick mark 1/2 a bar width and the other bar group to the left/below the tick mark 1/2 a bar width.
3 bar groups:	center 1 bar group over the tick mark; displace 1 bar group to the right/above the tick mark 1 bar width and the other bar group to the left/below the tick mark 1 bar width.

The border of the bar should be the same color as the interior of the bar. Each bar group should have different colors.

The bar label should also have the same color as the bar.

DATA REPRESENTED ON PIE GRAPHS. EZDRAW allows the user to design pie graphs with up to ten segments. However, it is recommended that pie graphs be limited to 4 or 5 segments.

When the data is linked to the pie graph, EZDRAW by defaults places the pie in the center of the chart space and gives the pie a radius equal to 1/3 of the chart space in the Y-extent direction. The user may respecify these attributes whenever it is required. The user may also specify an initial rotation angle (from the horizontal) to where the first segment is drawn. The user may also specify whether the pie shall be drawn clockwise or counter-clockwise. (REF G:A-56)

Segments of the pie may be exploded for emphasis of these segments. The exploded segments must be contiguous. Different groups of contiguous segments may be exploded as long as the segments of each group are mutually exclusive.

1. A segment that has been exploded may not be exploded again.

2. The whole pie may not be exploded as one contiguous group.

3. Each segment of the pie may be exploded singly.

The explosion factor (distance of separation from the pie) is an EZDRAW standard equal to 25% of the pie radius.

Each pie segment has its own attributes that are user specified. The segment should have the same color for the border and interior. Segments should have different colors and/or different interior patterns. (REF G:A-61)

Each segment will have its own label specified by the user. It should be the same color as the segment if possible. The label is positioned outside of the segment by default. Justification is ignored since the label is positioned automatically based on the position of the segment.

The user has the option of having the label, a percentage, and the relative value of the segment displayed singly or in any combination of the three. All acquire the attributes of the text label. The percentage and segment values are placed below the label by default. Once the option to display either the percentage or segment value has been chosen, these values are displayed on all segments and can not be removed. (REF G:A-63, A-64, A-66)

RECOMMENDATIONS

Since EZDRAW is a new system, the following sequences for drawing graphs are provided. As a result of EZDRAW testing, attempting to draw the graph in a sequence other than that recommended results in unnecessary delays.

This is due to the fact that most graph elements are related to the data. If an attempt is made to specify data related elements before the data is linked (i.e. a user attempts to specify a curve before the data for the curves have been linked to the graph) then non-fatal error messages will result.

GRAPH DRAWING SEQUENCES

LINE GRAPHS. The following sequence is recommended for drawing line graphs.

1. Link the graph data to the graph:

   Indicate the start and end values for the independent variable axis,
   Indicate the start and end values for the dependent variable axis, and
   Designate which axis is to be the independent variable axis.

2. Specify the attributes for the independent axis then the dependent axis.

3. Specify tick marks (and attributes) along the independent axis and tick marks (and attributes) along the dependent axis.

4. Specify curve 1, then curve 2, and then curve 3.

5. Specify the label for the independent axis and position the label.

6. Specify the label for the dependent axis and position the label.

7. Specify the label group for the tick marks along the independent axis.

8. Specify the label group for the tick marks along the dependent axis.

9. Specify the label for curve 1 and position it; then do the same for curve 2 and then for curve 3.

10. Specify the main title for the graph and position it; then do the same for sub-title 1 and then sub-title 2.

11. Specify additional note 1 and position it; then do the same for additional note 2 and additional note 3.

BAR GRAPHS. The same sequence as noted for line graphs above should be followed. Items 4 and 9 should appropriately be changed to the specification of bars and bar labels.

PIE GRAPHS. The following procedure should be followed for designing pie graphs.

1. Link the data to the pie graph.

2. Specify the position of the pie, radius, direction, and rotation of the pie (can be done as often as desired).

3. Define the segments of the graph. It is simpler to start with segment number 1 and then continue in sequence until all segments have been specified.

4. Define the segment labels. Again, it is simpler to start with the first segment and continue in sequence until all segment labels have been specified.

5. Explode any pie segment(s) desired.

6. Same as Item 10 in Line Graph.

7. Same as Item 11 in Line Graph.

VITA

Roldan Vea was born in Bacarra Ilocos Norte, Republic of the Philippines on 9 February 1951. He graduated from high school in El Paso, Texas in 1969 and then attended the United States Military Academy, West Point, New York where he was commissioned and received a Bachelor of Science degree in Applied Science and Engineering in 1973. After attendance at the US Army Field Artillery School at Fort Sill, Oklahoma he served as Battery Reconnaissance/Survey Platoon Leader, Battery Fire Direction Officer, and Assistant Battalion Intelligence/Security (S2) Officer with the 1st Battalion, 27th Field Artillery at Fort Carson, Colorado. In 1975, he transferred to the Adjutant General Corps and was assigned to the Headquarters staff of the 4th Infantry Division at Fort Carson. He next served with the V Corps Personnel and Administration Battalion in Hanau, Germany as the Chief, Personnel Records Division and in Frankfurt, Germany as the Battalion Intelligence/Security and Operations/Training (S2 & S3) Officer. He was then assigned to Fort Benjamin Harrison, Indiana where he attended the Adjutant General Officer Career Course, completed the Automated Data Processing Officer Course, and served on the staff of the US Army Computer Science School. He entered the Air Force Institute of Technology in June, 1981.

Permanent Address: 5707 Bainbridge Avenue
El Paso, Texas 79924

UNCLASSIFIED
SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

REPORT DOCUMENTATION PAGE		READ INSTRUCTIONS BEFORE COMPLETING FORM
1. REPORT NUMBER AFIT/GE/MA/82D-1	2. GOVT ACCESSION NO. AD-A124 907	3. RECIPIENT'S CATALOG NUMBER
4. TITLE (and Subtitle) EZDRAW-AN INTERACTIVE COMPUTER GRAPHICS PROGRAM TO DESIGN BAR, LINE, OR PIE GRAPHS		5. TYPE OF REPORT & PERIOD COVERED MS THESIS 6. PERFORMING ORG. REPORT NUMBER
7. AUTHOR(s) Roldan Vea CPT, USA		8. CONTRACT OR GRANT NUMBER(s)
9. PERFORMING ORGANIZATION NAME AND ADDRESS Student, HQDA, MILPERCEN (DAPC-OPP-E) 200 Stovall Street Alexandria, VA 22332		10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS
11. CONTROLLING OFFICE NAME AND ADDRESS HQDA, MILPERCEN, ATTN: DAPC-OPP-E 200 Stovall Street Alexandria, VA 22332		12. REPORT DATE December 1982 13. NUMBER OF PAGES 227
14. MONITORING AGENCY NAME & ADDRESS(if different from Controlling Office) Air Force Wright Aeronautical Laboratories ATTN: AAWA WPAFB, OH 45433		15. SECURITY CLASS. (of this report) Unclassified 15a. DECLASSIFICATION/DOWNGRADING SCHEDULE

16. DISTRIBUTION STATEMENT (of this Report)

Approved for public release; distribution unlimited.

17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)

18. SUPPLEMENTARY NOTES

Approved for public release: IAW AFR 190-17
LYNN E. WOLAVER
Dean for Research and Professional Development
Air Force Institute of Technology (AU)
Wright-Patterson AFB OH 45433

19. KEY WORDS (Continue on reverse side if necessary and identify by block number)

Interactive Computer Graphics
Structured Analysis and Design
Graphics Design

20. ABSTRACT (Continue on reverse side if necessary and identify by block number)

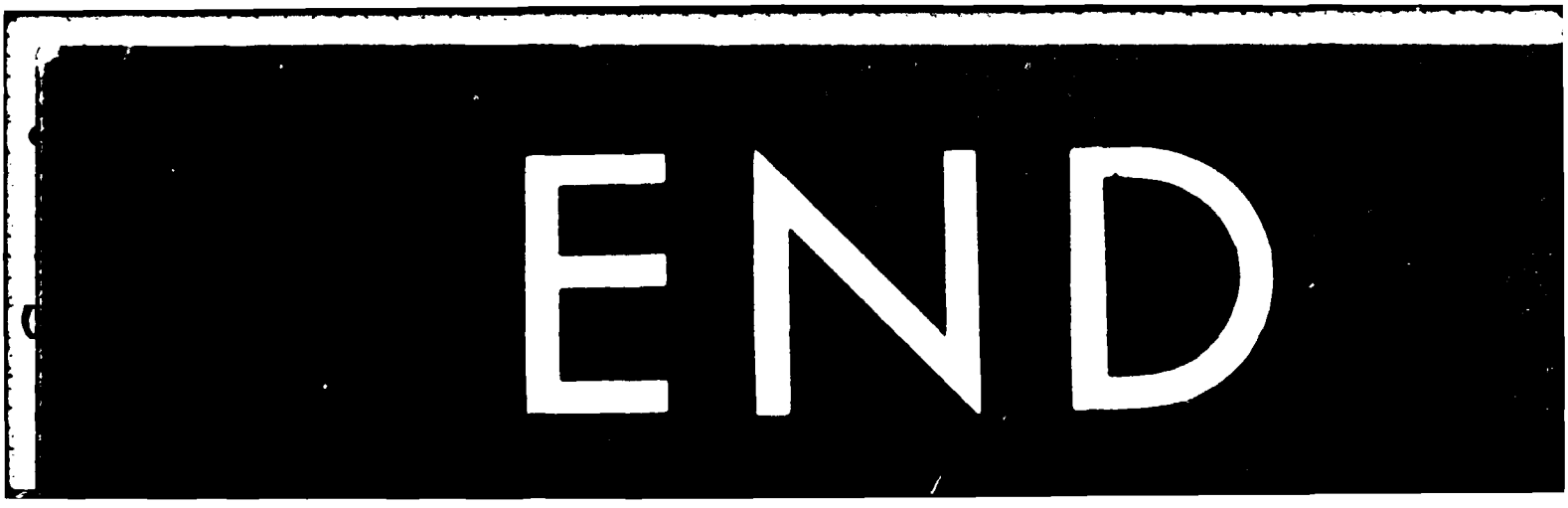
This study initiates the development of an interactive computer graphics system. This included the integration of off-the-shelf CORE standard graphics software for use on several computer graphics refresh and raster display devices. The system was implemented on a Digital Equipment Corporation VAX 11/780 mini-computer in standard FORTRAN 77. The system's requirements and top-level design specifications were generated following a

DD FORM 1 JAN 73 1473 EDITION OF 1 NOV 65 IS OBSOLETE

UNCLASSIFIED
SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

20. ABSTRACT

study of graphics design concepts. The operational requirements of Precision Visuals' DI-3000 and GRAFMAKER software packages were interfaced with the system requirements to obtain the system level design specifications. The concepts of good interactive man-machine communication were included in the final design. This produced a reliable graphics system responsive to designers of bar, line, or pie graphs. The system was designed using structured analysis and design techniques which will allow a follow-on study to implement recommended modifications or enhancements from the supporting graphics software.


END